Université de Toulouse

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par *l'Université Toulouse III - Paul Sabatier*

Discipline ou spécialité : *Informatique*

Présentée et soutenue par

Hassan WEHBE

Le lundi 20 juillet 2016

SYNCHRONISATION AUTOMATIQUE D'UN CONTENU AUDIOVISUEL AVEC UN TEXTE QUI LE DÉCRIT

Membres du JURY

Directeur de Thèse : Philippe JOLY, professeur à l'université Paul Sabatier (UPS)
Co-directeur de Thèse : Bassem HAIDAR, professeur associé à l'université Libanaise
Rapporteur : Serge MIGUET, professeur à l'université lumière Lyon 2
Rapporteur : Georges QUENOT, directeur de recherche au CNRS (équipe MRIM)
Examinateur : Michel DEVY, directeur de recherche au CNRS (LAAS)
Examinatrice : Ewa KIJAK, Maitre de conférence et chercheuse à l'IRISA - TEXMEX

École doctorale et discipline ou spécialité : *ED MITT : Image, Information, Hypermedia*

Unité de recherche : IRIT - SAMoVA

... à mes parents Ali et Souad,
la source du bonheur
qui ont fait l'impossible pour
nous rendre heureux et
réaliser nos et leurs rêves

... à ma femme Zainab que je n'imagine pas
ma vie sans elle

"Soyons reconnaissants aux personnes qui nous donnent du bonheur ; elles sont les charmants jardiniers par qui nos âmes sont fleuries."

De "Marcel Proust"

REMERCIEMENTS

Je tiens à exprimer mes plus vifs remerciements à mon directeur de thèse, M. Philippe JOLY, qui fut pour moi un directeur de thèse attentif et disponible malgré ses nombreuses charges et responsabilités. Merci pour m'avoir apporté l'aide, le soutien, la motivation dont j'avais besoin à chaque moment aux niveaux professionnels et personnels. Sache que vous m'avez aidé bien plus que vous ne le pensez alors je vous écris un "Merci" qui vient vraiment du fond du cœur. J'étais chanceux de vous reconnaitre, une personne gentille, polie, amicale, serviable et respectueuse.

Mes sentiments de reconnaissance et de gratitude les plus profonds vont pour M. Bassem HAIDAR, qui était plus qu'un co-directeur mais un ami et un grand frère qui s'intéresse toujours à moi et à ma progression. Je lui remercie d'être lui-même, une personne amicale, optimiste, encourageante, et philanthrope. Il avait toujours les conseils et les réponses particuliers.

Mes remerciements s'adressent à M. Serge MIGUET et M. Georges QUENOT pour avoir accepté d'être les rapporteurs de ma thèse. Ils m'ont fait honneur d'être aussi parmi le jury pour juger mon travail et me proposer les commentaires appréciés. Je voudrai remercie de même M. Michel DEVY pour avoir accepté d'être le président du jury, et aussi Mme Ewa KIJAK pour avoir accepté d'être une examinatrice dans ce jury. J'apprécie leurs présences et contributions pour rendre ma présentation plus efficace et intéressante.

Mes remerciements vont également vers la personne que je respecte pour ses collaborations appréciées, M. Bilal CHEBARO, qui sans lui je ne suis pas à cette situation. Je remercie aussi Mme Siba HAIDAR pour sa motivation et la confiance qu'elle m'a donnée durant ce projet pénible. J'adresse toute ma gratitude à mes amis (es) à l'Université Libanaise qui étaient toujours compréhensifs et encourageants, et à celui qui a redéfini le sens de l'amitié Hassan CHOUAIB.

Enfin, les mots les plus simples étant les plus forts, j'adresse toute mon affection à ma famille, l'univers auquel j'appartiens, la source de tout ce qui est jolie et formidable dans le monde. A Maman et papa, mes parents exceptionnels, le cadeau le plus précieux d'Allah, à mes frères Hossein et Mohamad, et mes sœurs amusants Nour el-ein et Fatima, desquels je suis fier. Les sentiments d'amour, d'appréciation, et de remerciements vont bien sûr à ma femme et mon âme Zainab, avec laquelle je veux continuer ma vie, la personne qui a rendu ce travail pénible plus facile, et qui était compréhensive malgré les défis de la vie.

RÉSUMÉ

Nous abordons le problème de la synchronisation automatique d'un contenu audiovisuel avec une procédure textuelle qui le décrit. La stratégie consiste à extraire des informations sur la structure des deux contenus puis à les mettre en correspondance.

Nous proposons deux outils d'analyse vidéo qui extraient respectivement :

- → les limites des évènements d'intérêt à l'aide d'une méthode de quantification de type dictionnaire
- → les segments dans lesquels une action se répète en exploitant une méthode d'analyse fréquentielle : le YIN.

Ensuite, nous proposons un système de synchronisation qui fusionne les informations fournies par ces outils pour établir des associations entre les instructions textuelles et les segments vidéo correspondants. Une "Matrice de confiance" est construite et exploitée de manière récursive pour établir ces associations en regard de leur fiabilité.

Mots clés : *Analyse de vidéo, documents audiovisuels, synchronisation, répétition, segmentation.*

TABLE DES MATIÈRES

Chapitre 1

INTRODUCTION GÉNÉRALE

Le travail de cette thèse se situe dans le domaine de l'analyse automatique des contenus audiovisuels. Il cible la réalisation d'un système de synchronisation d'un contenu audiovisuel avec le texte procédural qui le décrit. Ce système traite un problème original qui pourra servir de base pour des travaux de synchronisation sur des contenus plus génériques ou au contraire spécialisés.

1.1 Sujet et motivation

L'analyse automatique des contenus multimédia est devenue une nécessité essentiellement dans un but d'économie des efforts humains. La plupart des travaux sur ce sujet ne concernent qu'un type unique de média (vidéo, audio ou texte) avec différents objectifs (indexation, structuration, détection des copies, etc…). Mais, rares sont les travaux qui traitent l'analyse en parallèle de deux médias différents, comme cela est nécessaire pour la synchronisation d'un contenu vidéo avec un contenu textuel qui en décrit le déroulement.

Ce problème a déjà fait l'objet de propositions de solutions scientifiques dans un domaine applicatif particulier : l'alignement de scripts. Ces solutions reposent sur des méthodes d'interpolation combinées à des technologies de reconnaissance de la parole. Pour l'heure, le problème plus vaste visant à établir des correspondances entre

un contenu audiovisuel et un texte qui le décrit (autrement que par le moyen d'une transcription littérale intermédiaire de la parole prononcée) reste ouvert.

L'objet de cette thèse est de proposer des moyens d'analyse des informations audiovisuelles dans le but d'établir des mises en correspondance avec le contenu textuel. De manière à préciser l'objet de l'étude, les contenus étudiés concernent des textes procéduraux décrivant des séquences d'actions à réaliser (du type recette de cuisine, méthode pour le bricolage ou le jardinage, mode d'emploi pour le montage d'un meuble, procédure de maintenance d'un matériel industriel, etc...) et des enregistrements audiovisuels rendant compte de leurs exécutions.

Pour établir une synchronisation des contenus, nous proposerons des méthodes et des techniques pour délimiter les événements dans le contenu audiovisuel susceptibles d'être reliés à des instructions textuelles. Ces informations seront mises en correspondance avec une représentation sémantique dérivée de la prescription textuelle, par un système dédié.

Ce type des sujets trouve un champ d'application dans des domaines variés, comme la maintenance technique, ou la mise en ligne de tutoriels. Par exemple dans le domaine de maintenance, l'enregistrement vidéo d'une tâche spécifique réalisée par un technicien peut être confronté à la procédure pour identifier des erreurs ou un savoir-faire qui ne figure pas dans le texte.

1.2 Plan de la thèse

La stratégie proposée pour traiter le problème de synchronisation repose sur deux étapes : tout d'abord, nous détectons les limites des événements d'intérêt dans le contenu audiovisuel, puis nous mettons ces limites en correspondance avec les instructions dans le texte.

Dans le chapitre suivant, nous présenterons d'une manière détaillée l'objectif du travail et de manière générale la méthode de synchronisation. Nous présenterons les sujets les plus proches du nôtre sur le principe de mise en correspondance de deux contenus. Nous relèguerons la présentation des travaux liés aux méthodes qui seront proposées aux chapitres concernés. Ensuite, nous définirons le contexte du travail et les hypothèses que nous avons dû formuler pour pouvoir développer nos contributions. À la fin du chapitre, nous discuterons brièvement de l'analyse du contenu textuel, cette dernière n'étant pas traitée dans cette thèse.

Dans le chapitre "Segmentation des actions dans une vidéo", nous décrirons une méthode qui analyse la vidéo pour en extraire des points temporels correspondant aux limites d'événements d'intérêt. Cette méthode sera présentée et évaluée pour démontrer son efficacité et préciser sa performance.

Une seconde méthode sera proposée dans le chapitre suivant portant sur la détection des répétitions. Ce chapitre suivra le même plan que le précédent à travers

une présentation des travaux liés, la description de la contribution, et l'évaluation des résultats.

Une proposition d'un système de synchronisation sera introduite au chapitre "Synchronisation". Dans ce chapitre, nous proposerons des représentations et des techniques qui utilisent les informations fournies par les deux outils précédemment proposés pour établir la synchronisation. À la fin de ce chapitre, Nous verrons comment chaque instruction textuelle pourra être mise en correspondance avec le segment vidéo qui délimite son exécution. Une méthode d'évaluation sera appliquée sur un corpus de test. Ce chapitre fera l'objet d'une proposition d'éléments prospectifs.

Finalement, une conclusion générale relatant le processus de synchronisation proposé dans cette thèse, sera exposée. Nous évoquerons comment ce travail permet d'aborder de nouveaux sujets concernant plusieurs domaines d'applications.

Chapitre 2

CONTEXTE : ÉTAT DE L'ART, DÉFINITIONS ET HYPOTHÈSES

2.1 État de l'art général

Le système que nous nous proposons de réaliser repose sur plusieurs étapes. À chaque étape, nous abordons un sous-problème différent dans le but de produire des informations qui alimentent le système de synchronisation. Afin de donner de la cohérence à la présentation du système, l'état de l'art de chaque sous-problème est présenté dans le chapitre correspondant.

En raison de l'originalité du problème général abordé dans cette thèse, nous n'avons pas trouvé de travaux référencés qui traitent du même sujet. Cependant, nous présentons brièvement quelques travaux qui ont une certaine proximité avec les nôtres. Ceux-ci traitent le problème de la synchronisation de deux contenus différents ou l'analyse simultanée de deux contenus pour des raisons d'indexation.

La synchronisation d'un signal audio avec le script qui contient la version écrite des dialogues (appelée "**Word Spotting**") a fait l'objet de nombreux travaux (30), (31), (32). La première étape de la solution proposée en général est la reconnaissance de la parole dans le signal audio. Cette tâche est capable de fournir – avec un certain taux d'erreur – le texte prononcé dans le signal audio. La synchronisation est réalisée en établissant une correspondance entre les mots/phrases extraits automatiquement de la

bande son et les mots/phrases rédigés dans le contenu textuel. Au final, cette synchronisation est principalement réalisée de texte à texte.

D'autres travaux (comme (28) et (29)) abordent le même problème que les travaux ci-dessus mais d'une manière adaptée à un contenu spécifique. Les auteurs cherchent à synchroniser une vidéo d'une personne présentant un exposé avec support numérique qui accompagne cette présentation (un support PowerPoint, par exemple, contenant le texte présenté). Cette fois, les travaux intègrent également la détection du texte affiché dans la vidéo en plus de la parole dans le signal audio. De même, la synchronisation dans ce cas consiste en une mise en correspondance du texte extrait des deux contenus (support numérique et enregistrement audiovisuel). Par contre, les deux contenus ciblés par notre travail sont susceptibles de porter des informations de nature et de contenu différents. Le problème de mise en correspondance posé s'éloigne alors d'un problème de synchronisation.

D'autres travaux proposent d'analyser un contenu spécifique dans un flux TV en exploitant un scénario textuel prédéfini. (27) Utilise le scénario et les sous-titres pour annoter le contenu vidéo suite à la détection des scènes et des plans. (37) propose aussi une méthode pour annoter et structurer un long flux télévisé en utilisant le guide textuel des programmes. Dans le même esprit, de nombreux autres travaux ((33), (34), (35), (36)) traitent des problèmes spécifiques liés à l'exploitation conjointe de deux types de contenu (texte et vidéo correspondant) en s'appuyant principalement sur la structure de la vidéo analysée automatiquement (changement de scènes, de plans etc…) dans le but d'indexer la vidéo. Bien que leurs préoccupations semblent proches des nôtres, ces travaux abordent des problèmes très différents de ceux qu'on cherche à résoudre : nous analysons et extrayons les actions effectuées dans une vidéo (sans montage), et nous utilisons ces informations pour associer ce contenu vidéo avec l'instruction textuelle correspondante dans le texte.

2.2 Objectif

On considère dans ce travail deux contenus : un enregistrement vidéo et un contenu textuel. Le contenu textuel consiste à une liste d'instructions à suivre pour réaliser une tâche spécifique, et le contenu vidéo montre un opérateur unique qui exécute fidèlement ces instructions d'une manière consécutive et séparée (en d'autres termes, deux instructions ne sont pas réalisées en parallèle). La capacité de "**synchroniser**" ces deux contenus permet, par exemple, de vérifier si l'opérateur a bien suivi les instructions ou non. On cherche dans ce travail à proposer des outils et des méthodes permettant d'associer les instructions textuelles aux segments vidéo dans lesquels elles sont exécutées.

Ce travail repose sur un type de corpus spécifique qui contient des vidéos créées en respectant les hypothèses suivantes :

→ Les vidéos contiennent un seul opérateur qui agit sur son environnement en suivant une liste définie d'instructions textuelles,

→ Les vidéos doivent être enregistrées par une caméra unique et stationnaire

Nous n'avons pas trouvé de corpus de référence de ce type. Nous avons donc créé notre propre corpus pour des raisons de test et de vérification. Ce corpus consiste en des enregistrements vidéo courts qui respectent nos hypothèses. Nous gardons pour l'heure en perspective l'idée de pouvoir adapter les outils et les systèmes proposés au traitement de longs flux vidéo.

Notre stratégie consiste à extraire des informations structurées à partir de la vidéo et à mettre en place une méthode pour associer les instructions aux segments vidéo correspondants. À cette fin, nous devons préciser les types d'information qui serviront de support pour cette mise en correspondance.

2.3 Plan de travail

En général, la synchronisation entre deux contenus peut être définie comme un processus de mise en correspondance entre les informations du premier contenu et celles du deuxième. En travaillant avec des contenus temporels, ce processus consiste à associer un point sur la ligne de temps du premier avec le point correspondant sur la ligne de temps du deuxième. Nous désignons par contenu temporel, un contenu où une information donnée est identifiée par l'instant où elle apparaît, comme dans le cas d'une vidéo dont les événements sont repérables par leurs positions temporelles dans la vidéo. À l'inverse, un contenu non temporel est un contenu dont le déroulement est indépendant du temps, comme le cas d'un contenu textuel.

Définition : une instruction est une phrase qui demande à l'opérateur d'effectuer une action dans le cadre de la scène.

Pour mener à bien cette tâche, nous aurons à délimiter les instructions textuelles qui exigent l'exécution d'une action, à extraire les segments vidéo qui contiennent exclusivement une action, puis à exploiter les résultats pour synchroniser les deux contenus. Nous traiterons le problème de synchronisation de la manière suivante :

1) *Représenter les informations extraites de chaque contenu de manière structurée,*

2) *Mettre en place une méthode qui analyse chaque type d'information structurée dans les deux contenus, et lie chaque instruction au segment vidéo correspondant.*

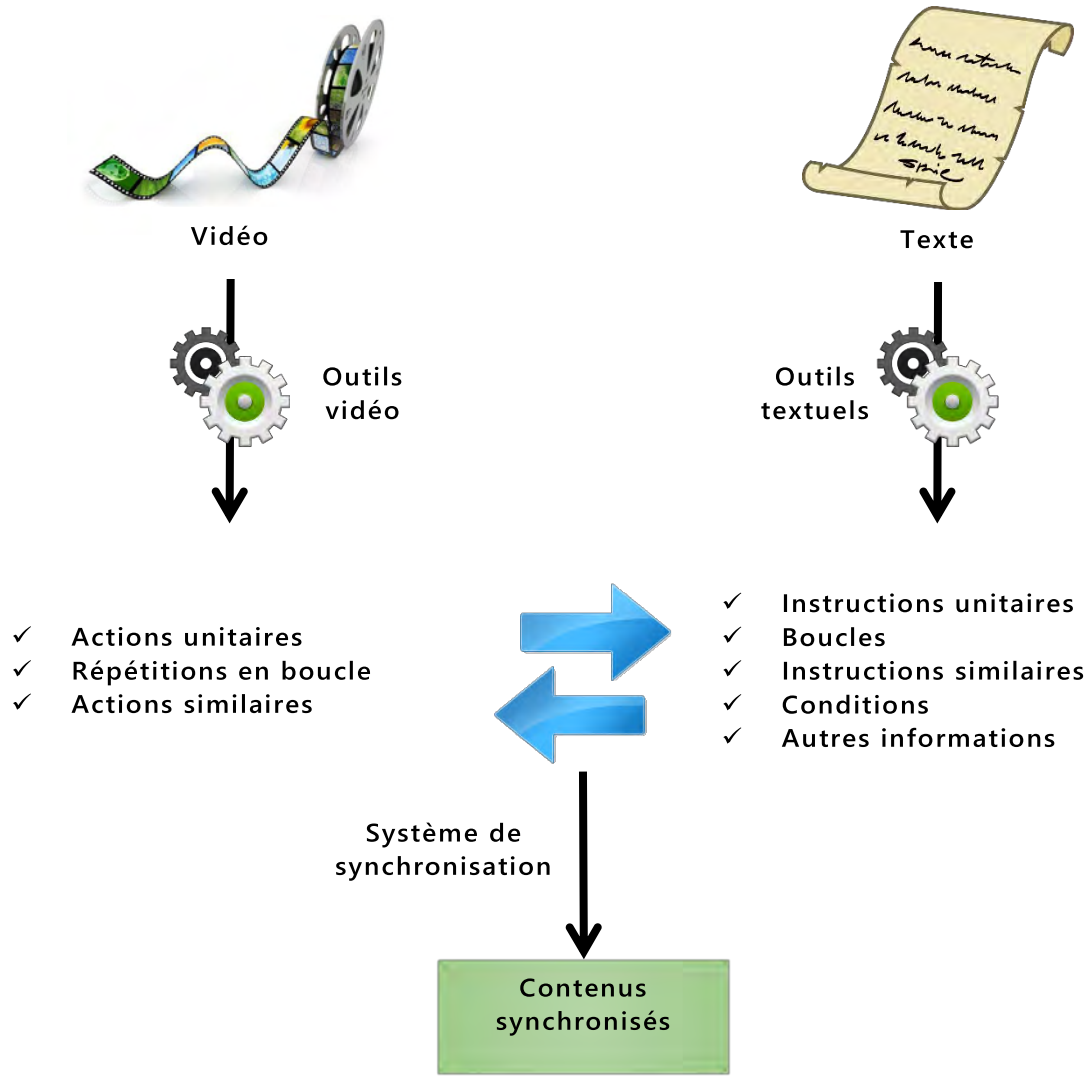


Figure 1: Schéma de synchronisation. Le type des informations présentées dans ce graphique sera défini plus tard.

2.4 L'analyse du texte

Le domaine de l'analyse automatique du texte regroupe aujourd'hui de nombreuses méthodes et outils qui visent à détecter et à extraire des informations d'intérêt à partir d'un texte rédigé en langage naturel. Un des nombreux champs applicatifs de ces méthodes est l'indexation des documents textuels dans de grandes bases de données en fonction du type de leurs contenus. Actuellement, ces outils sont très répandus et utilisables à travers les services proposés par Apple ou Google pour la gestion des emails. Les données textuelles sont ainsi partiellement analysées et interprétées pour trouver par exemple une date (d'un entretien, d'un événement, etc…), une réservation de billet d'avion, un numéro d'expédition, etc… et par suite activer une fonctionnalité adaptée ou orienter une communication commerciale de manière personnalisée.

2.4.1 Restriction

Notre travail repose principalement sur l'analyse du contenu vidéo. Au cours de cette thèse, nous allons proposer des outils pour extraire des informations à partir de ce contenu. Par contre, l'analyse du texte - nécessaire en théorie pour atteindre l'objectif fixé - ne fait pas partie de notre travail de recherche. Nous allons donc nous baser sur des hypothèses concernant les résultats attendus de l'analyse du texte pour la suite.

Dans cette section, nous ne prétendons donc pas présenter et encore moins proposer des outils d'analyse textuels. Ce n'est pas l'objet principal de notre contribution, et cela requiert des compétences propres à cette discipline. Nous discuterons dans ce qui suit de la forme générale d'un contenu textuel et des types possibles d'instructions que nous souhaiterions pourvoir identifier. Sur la base de l'hypothèse de l'existence d'un outil d'analyse du texte adapté, nous présentons le format des informations que nous souhaitons obtenir en guise de résultat de ce traitement pour poursuivre le processus de synchronisation, en soulignant les problèmes que nous pouvons juger a priori comme étant complexes à résoudre.

2.4.2 Les Informations textuelles requises

Le contenu textuel est une liste d'instructions qui vise à guider la réalisation d'une tâche spécifique. Chaque instruction exige l'exécution d'une ou plusieurs actions. La vidéo contient un ensemble d'actions effectuées à des moments inconnus, et potentiellement séparées par des contenus qui n'ont rien à voir avec les instructions textuelles. Afin d'associer chaque instruction au segment qui correspond à son exécution, nous devons d'abord extraire les segments vidéo qui délimitent l'exécution d'une instruction.

Définition : *Le contenu textuel est une liste d'instructions rédigées en langage naturel. Chaque instruction est une phrase qui demande l'exécution d'une ou plusieurs actions.*

À la suite dans cette section, nous présentons les éléments d'information relatifs à la structure que nous souhaitons extraire. Nous présenterons dans les chapitres ultérieurs des outils vidéo extrayant ces informations pour les utiliser dans le système de synchronisation. Toute information extraite des deux contenus peut jouer un rôle important dans le processus de synchronisation.

Nous considèrerons par la suite les types suivants :

2.4.2.1 Instruction unitaire

Nous supposons que dans le contenu vidéo, un opérateur humain exécute les instructions décrites dans le texte d'une manière consécutive. Cela suppose que cet opérateur interprète et transforme la liste des instructions textuelles en une liste d'actions consécutives. Le problème est donc de trouver une représentation textuelle décrivant les instructions à réaliser avec un niveau de granularité qui corresponde à celui des actions telles que nous serons susceptibles de les segmenter dans la vidéo. Pour représenter ce niveau de granularité, nous parlerons d'"**instruction unitaire**" que nous pourrons associer à une "**action unitaire**".

Définition : Une "Instruction unitaire" est une instruction qui demande l'exécution d'une action simple, cette action est alors dite "Action unitaire".

Par exemple, l'instruction "Accrocher une photo sur le mur" consiste à effectuer plusieurs actions ("Tenir un clou ; l'enfoncer dans le mur, prendre la photo et l'accrocher sur le clou"). Par suite, ce n'est pas une instruction unitaire. Par contre, l'instruction "enfoncer un clou dans le mur" est considérée comme unitaire.

Dans la mesure où la définition de ce que peut être une action unitaire est propre aux travaux de cette thèse, et où la notion d'instruction unitaire est directement corrélée à cette première définition, des travaux ad' hoc dans le domaine de l'analyse automatique du texte seraient nécessaires pour pouvoir assurer la mise en correspondance entre ces deux unités. Nous pouvons gager que la transcription d'une procédure textuelle sous forme d'une séquence d'instructions unitaires est un problème riche et complexe qui devrait être l'objet de travaux de recherche spécifiques. Les hypothèses correspondant à ce point seront formulées plus tard dans ce chapitre.

La synchronisation de ce type d'informations consiste alors en la mise en correspondance de chaque instruction unitaire avec le segment vidéo contenant l'action unitaire correspondante.

2.4.2.2 Boucle

Parfois, le contenu textuel contient des instructions qui doivent être répétées plusieurs fois de manière consécutive (répétition en boucle). L'exécution de cette boucle exige la répétition d'une ou plusieurs action(s). Pour cela, nous chercherons à associer les boucles du texte aux segments vidéo qui contiennent une répétition, une fois leur détection réalisée sur les deux médias.

2.4.2.3 Condition

Comme dans le cas d'un algorithme, la procédure textuelle contient parfois des instructions dont l'exécution dépend de l'évaluation d'une condition spécifique. De ce fait, chaque exécution de cette procédure peut correspondre à une réalisation

différente. Identifier ce type de structure est donc très important pour la synchronisation, de manière à pouvoir ignorer les instructions qui ne sont pas exécutées dans la vidéo.

2.4.2.4 Instruction répétée

Il est également possible d'avoir deux instructions qui exigent l'exécution de la même action dans la vidéo, mais à deux moments séparés dans le temps. Cela suppose qu'on observe deux segments vidéo contenant la même action. L'identification des instructions similaires dans le texte et les segments similaires dans la vidéo peut être un indice important pour la synchronisation.

2.4.2.5 Autres types

D'autres informations supplémentaires peuvent être aussi utiles pour affiner la synchronisation. Les plupart de ces informations sont d'origine textuelle, comme par exemple :

→ Des descripteurs numériques :

- Un nombre d'occurrence ou une durée estimée pour l'exécution d'une boucle dans le texte, par exemple, peut être très utile pour l'outil de détection des répétitions dans la vidéo, comme nous le verrons, pour optimiser le temps de calcul et pour affiner les résultats.
- Une durée estimée pour l'exécution d'une instruction peut permettre d'affiner la localisation des limites du segment correspondant.
- Un temps explicite de séparation entre l'exécution de deux instructions spécifiques affine aussi la sortie de la méthode de synchronisation.

→ Des descripteurs qualitatifs :

- Ce sont des termes textuels décrivant par exemple la vitesse d'exécution : "Rapidement", "doucement", etc... Dans certains cas, ces informations peuvent être utiles pour préciser une durée minimale (réalisation lente) ou maximale (réalisation rapide) du segment qui contient l'exécution de l'instruction correspondante.
- Des termes qui invitent l'opérateur à suspendre l'exécution des actions, avant de les relancer après un temps spécifique. Cela permet de définir une distance minimale entre l'exécution de l'instruction qui précède et celle qui suit l'arrêt.

2.4.3 Étude du contenu

Une analyse textuelle est alors nécessaire pour identifier dans le texte les types d'informations textuelles présentés ci-dessus. Ce type de traitement s'apparente à celui mis en œuvre pour assurer l'extraction et la mise en forme de manière structurée des informations à partir d'un texte à travers la reconnaissance des mots, des phrases, de leurs rôles grammaticaux, de leurs relations et de leurs sens.

"Le 20 Avril 1964, la société italienne Ferrero a créé la pâte à tartiner Nutella, la plus populaire dans le monde. "

Représentation 1:

Création (Produit, Producteur, Date) → Création (Nutella, Ferrero, 20 Avril 1964)

Représentation 2 :

En <date>20 Avril 1964</date>, la société italienne <entité>Ferrero</entité> a créé la pâte à tartiner <entité>Nutella</entité>, la plus populaire dans le monde."

Représentation 3 :

Type d'événement	**Création**
Date d'événement	**20 Avril 1964**
Producteur	**Ferrero**
Produit	**Nutella**
Type de produit	**pâte à tartiner**
Emplacement	**Italie**
Popularité	**très populaire**

Figure 2: Exemple d'extraction et de structuration d'informations à partir de l'interprétation automatique d'une phrase.

Voici de manière résumée les principaux sujets abordés par des travaux d'analyse et d'extraction d'informations du texte (72):

1) *La récupération des informations: est une étape préparatoire qui consiste à collecter et identifier les ressources textuelles à partir d'un fichier, web, base de données, etc…*

2) *La reconnaissance d'entités nommées regroupe des techniques permettant de distinguer les mots du texte tels que les noms des personnes, organisations, endroits, abréviations, etc…*

3) *La reconnaissance d'informations à format standard comme les numéros de téléphones, les adresses email, des unités de mesure, etc…*

4) *La coréférence : c'est le fait d'être capable à distinguer les entités qui désignent le même objet, comme de savoir quelle est l'entité désignée par un "il" à un certain endroit (ex. "Philippe est un enseignant, et il aime son métier").*

5) *La compréhension des relations entre les entités, comme la relation entre une personne et une organisation, entre une personne et un lieu, etc…*

6) *L'extraction des sentiments, des opinions, de l'humeur et de l'émotion.*

7) *L'extraction des relations sémantiques et grammaticales pour comprendre le sens entre les phrases et les mots*

8) *Etc…*

Cette liste montre la diversité des travaux dans ce domaine, et donne une idée de la difficulté de trouver une méthode générique qui soit susceptible d'interpréter et de transcrire les instructions rédigées en langage naturel, comme par exemple : une méthode pour transcrire le texte en une liste d'instructions unitaires, ou pour distinguer les instructions qui consistent en une répétition implicite, etc…. Afin de poursuivre la présentation de notre méthode de synchronisation, nous formulerons pour le contenu textuel, les hypothèses correspondantes.

2.4.4 Hypothèses de travail

Hypothèse I. *L'outil d'analyse textuel est capable de transcrire le contenu textuel sous la forme d'une séquence d'instructions unitaires correspondant aux actions susceptibles d'être segmentées automatiquement dans l'enregistrement vidéo.*

Dans le cas où le contenu textuel serait formulé dans un langage structuré et à l'aide de termes prédéfinis, la réalisation de l'Hypothèse I peut être effectuée d'une manière plus facile. Par contre, dans le cas du langage naturel, ce qui est notre cas, il est difficile et complexe de la réaliser en tenant compte de la diversité des formes possible du contenu. Cela conduit nécessairement à interpréter les sens des instructions avant de les transformer en une séquence d'instructions unitaires.

Par exemple, l'instruction "**Accrocher une photo sur le mur**" consiste à effectuer plusieurs actions ("**Tenir un clou ; l'enfoncer dans le mur, prendre la photo et l'accrocher sur le clou**"). L'outil textuel cherché est capable de transcrire l'instruction originale en une liste d'instructions unitaire.

Nous allons considérer maintenant qu'une procédure textuelle peut être transposée sous la forme d'un graphe dont les nœuds sont des instructions unitaires et les arêtes décrivent le sens et l'ordre d'exécution de ces instructions.

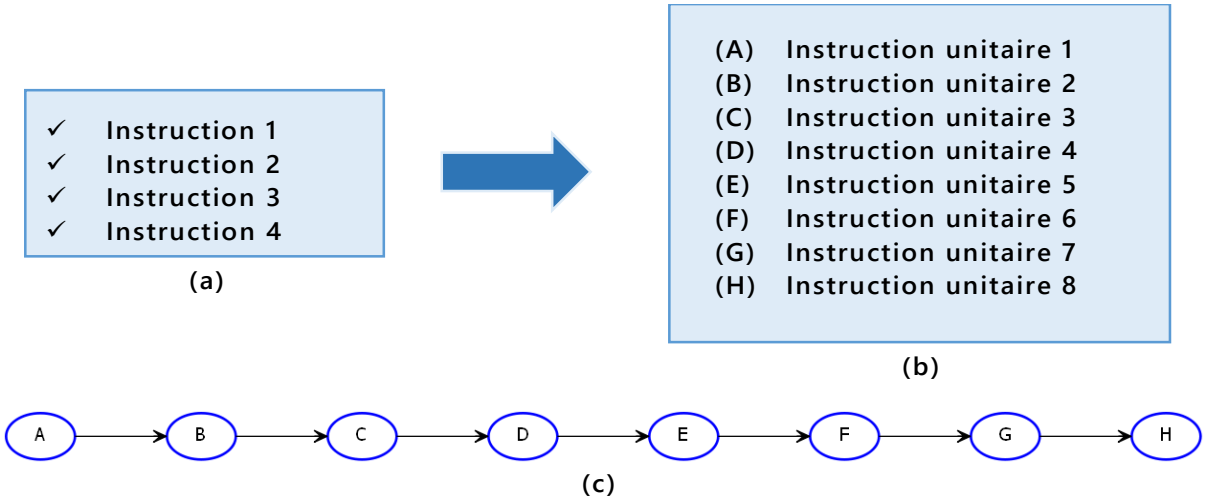


Figure 3 : Description du fonctionnement supposé de l'outil d'analyse textuelle. (a) est le contenu textuel original, (b) est la liste des instructions unitaires produite par l'outil supposé, et (c) est le graphe correspondant qu'on souhaite obtenir.

Dans la Figure 3, nous remarquons que certaines instructions du contenu original exigent l'exécution de plusieurs actions unitaires dans la vidéo. Selon l'Hypothèse I, l'outil hypothétique d'analyse du texte est capable d'interpréter ces instructions, et de transcrire le texte sous la forme d'un graphe où les sommets sont des instructions unitaires.

D'autre part, le contenu textuel n'est pas toujours constitué uniquement d'instructions simples. Il peut contenir des termes qui contrôlent la démarche d'exécution comme les termes qui indiquent une répétition (boucle) ou une condition à évaluer. Ce sont les structures de contrôle bien connues en algorithmique, qui rendent potentiellement le problème plus complexe (Figure 4).

Parfois l'expression de ces termes est standard, (*ex.* "**Répéter cette action dix fois**", "**Si la lampe est verte, alors …**", *etc*…), mais ce n'est pas toujours le cas et cette expression peut prendre des formes variées. Ces éléments peuvent disparaître dans une forme implicite des instructions (ex. une boucle de l'action "**frappez le clou avec le marteau**" dans "**Enfoncez le clou dans le mur**", une condition dans "**Vous souhaitez une pizza plus épicée ? Ajoutez un piment vert.**"). Cela illustre sur la complexité et la diversité des cas à traiter avec l'outil d'analyse textuelle. Dans la

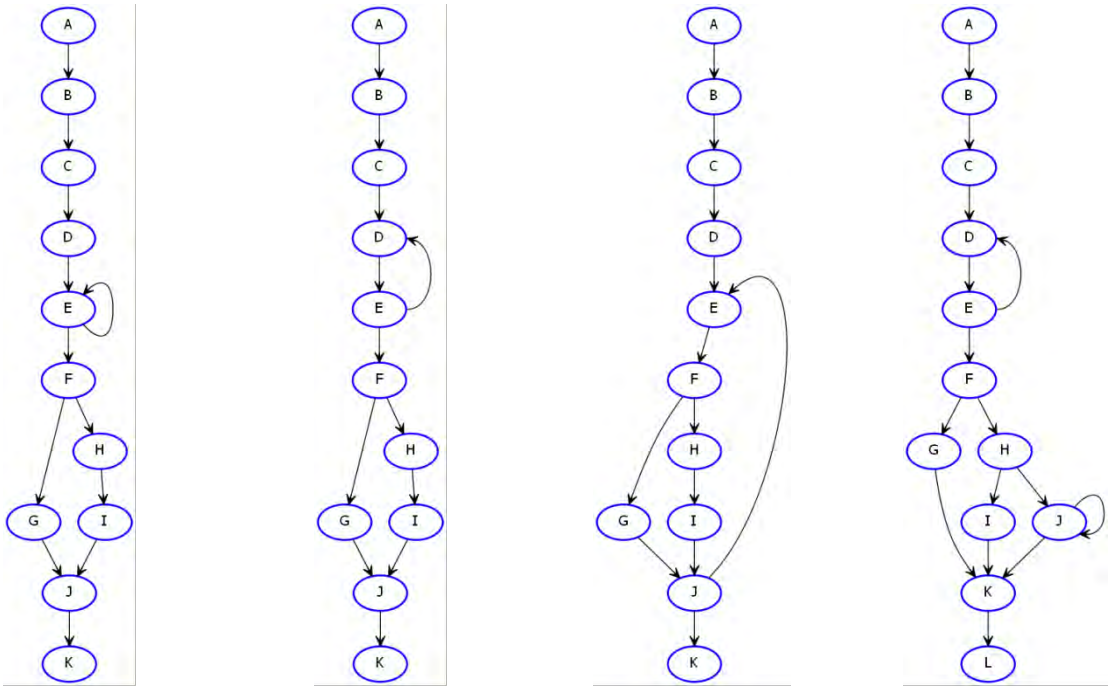
(A) Instruction unitaire 1
(B) Instruction unitaire 2
(C) Instruction unitaire 3
(D) Instruction unitaire 4
Début de boucle
(E) Instruction unitaire 5
Fin de boucle
(F) Instruction unitaire 6
Si <condition>
(G) Instruction unitaire 7
Sinon
(H) Instruction unitaire 8
(I) Instruction unitaire 9
Fin de condition
(J) Instruction unitaire 10
(K) Instruction unitaire 11

Figure 4 : Un contenu textuel à une démarche complexe.

mesure où ces informations seront importantes pour notre contribution, nous sommes amenés à formuler l'hypothèse suivante :

Hypothèse II. *Nous supposons que l'outil d'analyse textuelle est capable d'identifier les positions des boucles et des conditions dans le texte.*

Le résultat de cette analyse peut être intégré dans la représentation sous forme de graphe. La Figure 5 montre des exemples des graphes qui correspondent à des contenus textuels contenant des boucles et des conditions.



(a) (b) (c) (d)

Figure 5: Exemples de représentations graphiques de quatre contenus textuels. Dans ces exemples, les instructions situées dans des boucles sont représentées par une **arête** *orientée de la dernière instruction dans la boucle vers la première (ex. dans le graphe (b) : les instructions "D" et "E" sont situées dans une boucle). L'existence d'une condition après une instruction donnée, est indiquée par plusieurs* **arêtes** *qui partent de cette instruction (ex. après l'instruction "F", il existe une condition à évaluer).*

Le quatrième type d'informations qu'on imagine possible d'extraire du texte (et que nous exploiterons), est l'ensemble des instructions similaires. Dans notre étude, nous considèrerons que deux instructions unitaires sont similaires si elles exigent l'exécution de la même action unitaire. Tandis qu'il est possible de repérer les instructions formulées de la même façon (le même texte), on peut imaginer qu'il peut

être difficile de les distinguer si la même action est décrite de deux manière différentes dans les instructions textuelles, comme par exemple : "**Ajouter le sucre au mélange**" et "**Sucrer la préparation**". Nous formulons donc l'hypothèse suivante :

Hypothèse III. *On suppose que l'outil d'analyse textuelle est capable d'identifier les ensembles d'instructions similaires.*

Le dernier type d'informations à extraire correspond à un sujet fréquemment abordé par les travaux du domaine (*cf.* (73), (74), (75)). L'objectif est d'extraire des indicateurs temporels (*ex.* "**avant**", "**après**", "**doucement**", etc...), des valeurs numériques (*ex.* "**4 fois**", "**sept cuillères**", etc...), des indicateurs spatiaux (*ex.* "**au-dessus**", "**vers la gauche**", *etc*...). Nous formulons alors la dernière hypothèse sur le fonctionnement de l'outil d'analyse textuelle :

Hypothèse IV. *Nous supposons que l'outil d'analyse textuelle extrait les indicateurs temporels, numériques, spatiaux et d'autres informations quantitatives et qualitatives, si elles existent.*

2.4.5 Format attendu des informations

En résumé, nous attendons de l'outil d'analyse textuelle la production d'informations au format suivant :

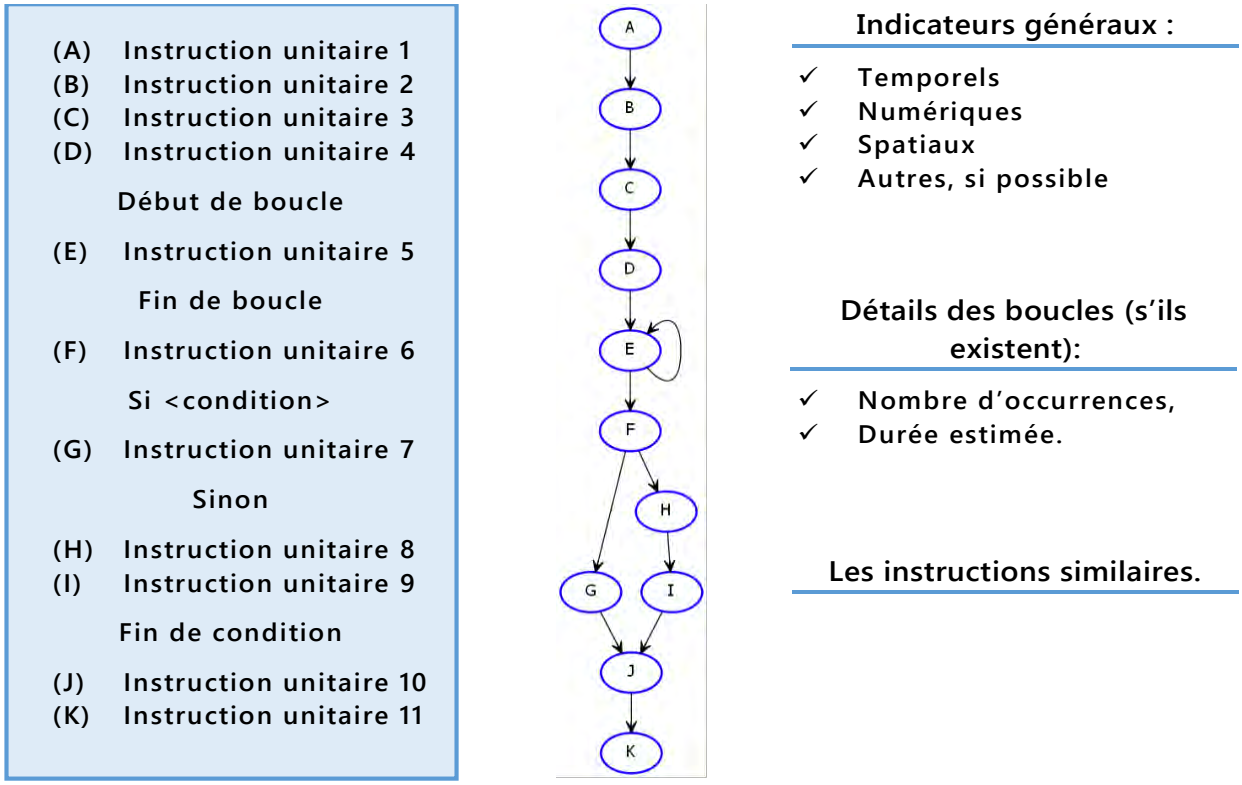


Figure 6: Le format souhaité des informations.

À partir de cette représentation, nous devons connaître :

→ Le nombre des actions unitaires, et leur ordre d'exécution,

→ Les instructions unitaires,

→ La position et les limites des boucles,

→ Les chemins d'exécution possibles,

→ Les instructions similaires,

→ Les informations supplémentaires (s'il y en a).

2.5 Verrous scientifiques

Les hypothèses formulées précédemment sont potentiellement autant de verrous scientifiques portant sur l'analyse du texte que nous supposons résolus. Dans notre travail, nous nous intéresserons plus précisément aux problèmes suivants :

→ Le repérage automatique du début et de la fin d'une quelconque action unitaire. Les difficultés majeures de ce problème consistent dans la définition de ce que peut être une action unitaire et dans la grande diversité des représentations visuelles de ces actions dans la vidéo.

→ Le repérage automatique des structures de boucle dans l'enregistrement vidéo. Ce type de structure n'apparaît que très rarement dans un enregistrement vidéo quelconque, mais trouve pleinement sa place dans le cas de l'exécution d'une

procédure. Par le fait, c'est un sujet qui n'a jamais été réellement identifié en tant que tel dans l'état de l'art.

→ L'identification de solutions pour détecter des ensembles d'actions similaires dans la vidéo. À l'inverse du problème précédent, il existe des travaux qui peuvent permettre d'y répondre dans une certaine mesure. Nous nous contenterons ici d'identifier une piste possible.

La synchronisation de deux contenus est assurée par la mise en correspondance entre les résultats supposés de l'analyse textuelle et ceux issus de l'analyse vidéo. En l'état, nous aurons là aussi à formuler une proposition originale car elle sera fortement dépendante des résultats des méthodes précédentes.

Chapitre 3

SEGMENTATION DES ACTIONS DANS UNE VIDÉO

3.1 Introduction

3.1.1 Problématique

Dans le but de synchroniser les instructions textuelles et les segments vidéo qui représentent leurs exécutions, on doit d'abord trouver les segments qui incluent l'exécution d'une instruction individuelle, puis les lier aux instructions correspondantes. Nous considèrerons qu'une instruction textuelle décrit une action unitaire exécutée dans la scène. Une action peut être définie comme étant une séquence de mouvements qui conduisent à réaliser une tâche spécifique. Par exemple, une instruction qui consiste à fermer la fenêtre d'une chambre, conduit la personne à se déplacer d'abord vers la fenêtre, puis à manœuvrer la poignée pour la fermer. Mais le plus souvent, une tâche peut être exécutée de différentes façons. Elle pourra être complétée selon un de plusieurs scénarios différents. Donc, la recherche d'une action n'est pas toujours possible par le moyen d'une recherche d'une séquence spécifique de mouvements, d'où la complexité de ce problème.

3.1.2 Objectifs

Dans notre cas, les actions sont divisées en deux classes :

Classe 1. Action qui agit sur la scène

Cette classe comprend les actions qui ont un effet persistant sur la scène (que nous désignerons par "Arrière-plan"), comme le fait de déplacer un objet d'une place à une autre. C'est cette classe d'action qui sera considérée au cours de ce chapitre.

Classe 2. Action qui n'a aucun effet sur la scène

La deuxième classe comprend les actions qui se déroulent dans le premier plan et qui n'affectent pas la scène comme, par exemple le fait de lever une main. L'analyse des actions de cette classe forme le noyau de nombreux travaux (détection des visages et des gestes, suivi de cible, etc…). Dans notre thèse, nous nous sommes intéressés seulement aux actions de cette classe qui mettent en œuvre une répétition. La détection de ces actions sera décrite plus tard (Chapitre 4)

Dans ce chapitre, nous proposons une méthode pour localiser les limites des actions qui modifient l'arrière-plan. Comme nous l'avons déjà mentionné, une action de cette classe ne peut pas être détectée en s'appuyant seulement sur la caractéristique de mouvement. La propriété fondamentale de ces actions est la production d'une modification persistante sur la scène. Par suite, la détection du moment où une modification est exercée sur l'arrière-plan peut indiquer le début ou la fin d'une action. Le système que nous proposons d'élaborer aura pour objectif de détecter automatiquement les limites des actions. Nous appellerons ces limites "**limites automatiques**" pour les distinguer de celles annotées à la main, et que nous appellerons "**limites réelles**".

Du point de vue de l'analyse automatique du contenu vidéo, une modification exercée sur l'arrière-plan se traduit par un changement des couleurs des pixels d'une région, partant d'une couleur initiale à une couleur finale. Par exemple, l'ajout d'un nouvel objet sur la scène originale modifie les couleurs d'un ensemble de pixels d'une manière durable. Cet ensemble de pixels est composé principalement de ceux qui représentent l'objet à sa nouvelle position dans la scène (Figure 7: les pixels représentant la chaise noire). Dans cet exemple, ces pixels voient leurs couleurs passer du blanc au noir. La détection du moment auquel ces changements deviennent définitifs indique – dans cet exemple – la fin de l'action.

D'autre part, durant l'ajout du nouvel objet, il existe un autre ensemble de pixels qui subissent un changement temporaire de leur couleur, mais qui, à la fin de l'action, retrouvent leur couleur originale. Ils correspondent au passage de l'objet ou de l'opérateur humain durant l'exécution de l'action (Figure 8).

Nous proposons d'aborder le problème de segmentation en nous inspirant d'une technique de quantification de la couleur des pixels originalement proposée par KIM *et*

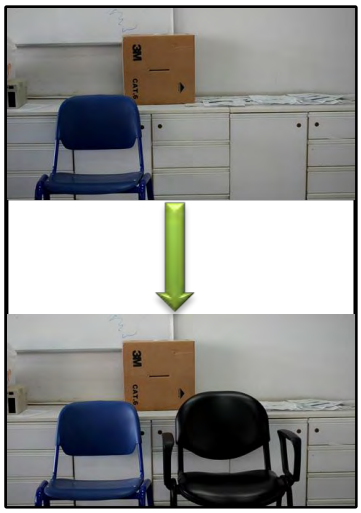


Figure 7: Action simple

Figure 8: Durant l'action

al. (45) pour construire un modèle de l'arrière-plan. Cette méthode sera adaptée et modifiée en regard de nos besoins qui diffèrent des objectifs initiaux de cette méthode. Ensuite, nous proposerons un automate de décision qui étudie et analyse les séquences des couleurs quantifiées des pixels. Le but de cet automate est de localiser les limites probables des actions de classe 1 au niveau de chaque pixel séparément. Les résultats sur l'ensemble des pixels alimenteront un système de votes visant à mettre en évidence les limites des actions dans la vidéo. Une méthode ad hoc sera finalement proposée pour extraire les positions temporelles de ces limites.

L'absence de corpus qui respectent les hypothèses de ce travail, nous a conduit à construire notre propre corpus pour pouvoir mener des expérimentations. Pour cette raison, nous ne serons pas en mesure d'appliquer notre méthode sur de longs flux vidéo. Mais l'outil que nous proposerons dans ce chapitre traite un enregistrement vidéo comme étant un flux, ce qui permettra d'étendre aisément la méthode à ce type de contenus ultérieurement.

3.1.3 Principe général de la méthode proposée

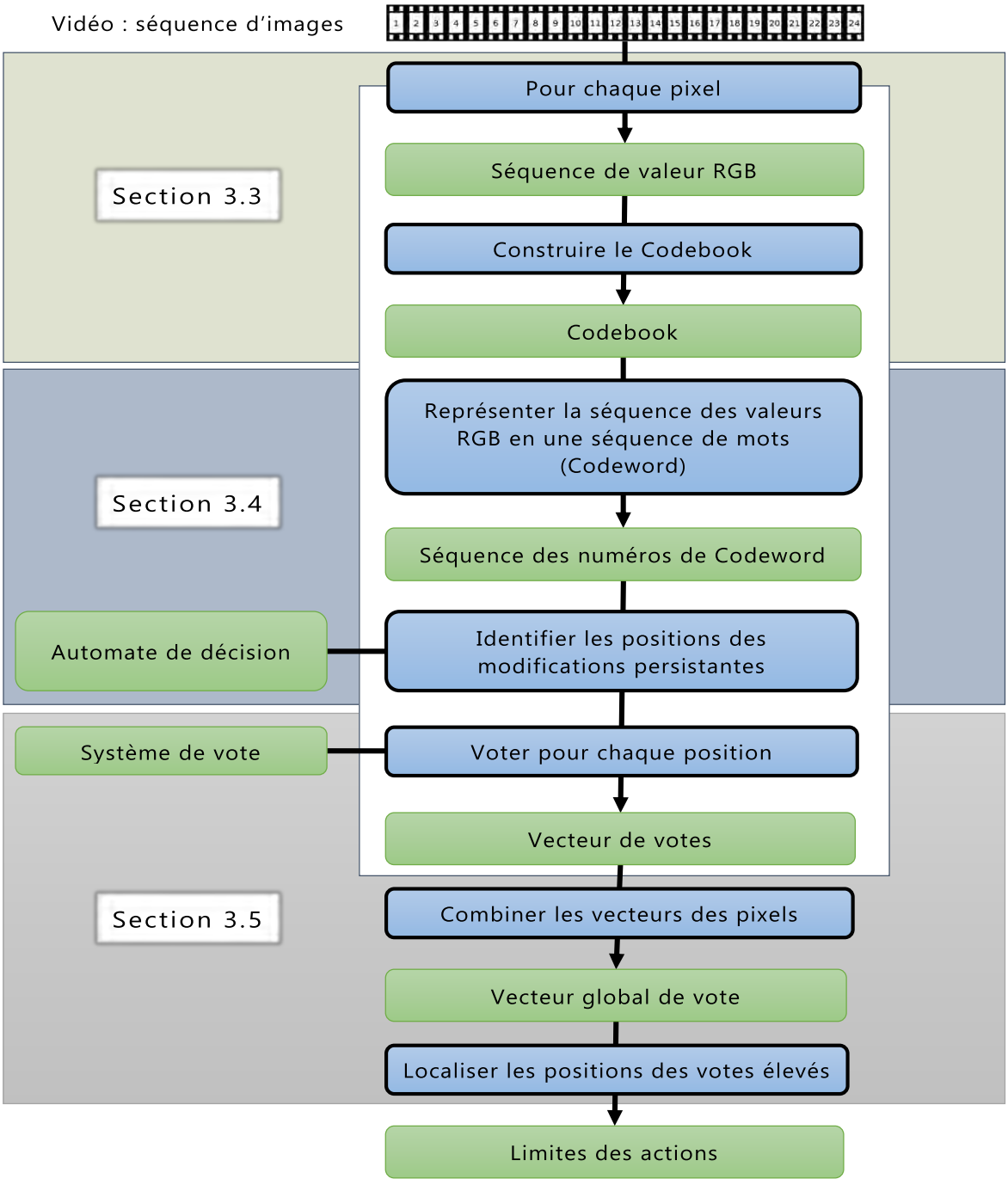


Figure 9: La démarche de la méthode de détection des limites.

3.2 État de l'art

3.2.1 Introduction

Dans ce chapitre, nous proposons un outil pour extraire les limites des événements d'intérêt (actions génériques et inconnues) dans une vidéo. Selon nos informations, la littérature scientifique ne propose pas de méthode pour délimiter des actions quelconques dans une vidéo. Nous présentons dans cette section quelques sujets traités dans différents articles en raison de leur proximité avec notre sujet (comme la détection d'événements) ou liés partiellement à notre méthode (comme la segmentation d'arrière-plan).

3.2.2 Détection des événements dans une retransmission sportive

La segmentation d'une vidéo vise à trouver les limites de segments qui contiennent une information d'intérêt spécifique. Dans notre cas, l'information d'intérêt est l'exécution d'une action exigée par une instruction textuelle unitaire. Les travaux les plus proches de ce sujet sont ceux qui localisent des événements précis dans une vidéo.

Certains des travaux qui traitent ce problème se concentrent sur l'extraction des événements importants dans les vidéos de jeux sportifs, comme la détection des buts dans un match de football. Certains se basent sur l'étude des caractéristiques visuelles en localisant la position de la balle à chaque trame pour structurer le déroulement d'un jeu de football (55), ou en détectant et suivant les joueurs d'un jeu de tennis pour trouver les moments importants (62). D'autres (59), identifient les événements importants dans un match de baseball en utilisant uniquement des caractéristiques audio. Ces événements sont localisés en étudiant des caractéristiques génériques des jeux sportifs et d'autres spécifiques au jeu de baseball. D'autres travaux préfèrent exploiter texte qui apparaît durant les jeux, comme les affichages de temps et des scores. Les auteurs de (64) ont analysé et étudié le contenu et le temps d'apparition des scores et des durées de jeu pour estimer les limites des événements d'intérêt dans une retransmission de baseball.

Ce type de travaux localise et identifie des événements prédéfinis dans des jeux sportifs spécifiques, ce qui n'est pas le cas dans notre travail. Dans ce chapitre nous présentons une méthode permettant de délimiter des événements (exécution d'instruction) génériques qui agissent sur la scène originale. Nous considérons des vidéos qui ne contiennent ni audio ni affichage textuel. De ce fait, le problème traité dans ce chapitre appelle des solutions potentiellement différentes des travaux qui traitent des enregistrements sportifs.

D'autres travaux dans le domaine de la détection d'évènements se concentrent sur la détection d'un comportement humain spécifique comme la détection d'une personne qui tombe par terre (67), la détection d'un comportement inhabituel (66) le non-respect du code de la route, les accidents, une conduite dangereuse, l'abandon

d'une valise (68), et d'autres encore ((69), (70), (71)). Dans ce type de travaux, les comportements à détecter sont connus a priori, mais dans notre cas, les actions à segmenter sont complètement inconnues.

3.2.3 Segmentation de l'arrière-plan

En général, les méthodes les plus populaires qui visent à segmenter l'arrière-plan construisent un modèle de l'arrière-plan (la scène originale) qui permet de le soustraire d'une scène future afin d'identifier les objets du premier plan (les objets mobiles). La valeur d'un pixel dans une nouvelle trame est comparée à la valeur correspondante dans le modèle, si elle est nouvelle alors ce pixel est considéré comme associé à un objet du premier plan (dans cette trame). La technique de base la plus utilisée pour modéliser l'arrière-plan est un modèle de mélanges gaussiens associé à l'intensité des pixels. Un modèle d'arrière-plan utilisant une distribution gaussienne unique présente des limites dans le cas où l'arrière-plan subit des changements majeurs (pluie, arbre dans le vent, *etc*...). D'autres méthodes plus efficaces sont dérivées de cette technique comme celles qui utilisent plusieurs distributions (61) ou des distributions non paramétriques ((45), (51), et (56)).

Les auteurs de (61) ont proposé d'utiliser le modèle de mélange gaussien pour chaque pixel au lieu d'utiliser la même distribution pour tous les pixels. Le problème de la prise en compte de l'ombre portée a été traité dans (63) par exemple en combinant cette technique avec une autre technique qui élimine l'ombre, ce qui a amélioré les résultats de modélisation et la soustraction d'arrière-plan.

Parmi les techniques à distributions non paramétriques, les utilisations fréquentes de la méthode proposée par (45) ((57), (50), (52)), prouvent que celle-ci est efficace pour modéliser et extraire l'arrière-plan. Selon les auteurs, cette méthode basée sur la technique de quantification par Codebook présente un avantage sur les méthodes basées sur la mixture de gaussiennes et les autres méthodes non paramétriques ((51)) en raison de sa capacité à traiter des flux. Chacun des travaux dérivés de cette technique ((56), (60), (53), (57), (50), (52)) modifie des caractéristiques de la méthode, soit en ajoutant des paramètres, ou en ajoutant un autre Codebook (60), ou bien en modifiant le processus de mise à jour du modèle pour améliorer sa performance (56) ou pour optimiser le calcul (53).

La littérature scientifique en analyse de vidéo est riche de travaux traitant le problème de la soustraction d'arrière-plan, mais notre contribution ne vise ni à comparer ces méthodes ni à proposer une nouvelle technique. Nous avons besoin d'une solution efficace pour quantifier toutes les valeurs dans l'historique des couleurs d'un pixel. Nous avons choisi d'utiliser la technique du Codebook proposée et utilisée dans (45) pour effectuer cette quantification en raison de son efficacité et de ses performances. Toutefois cette solution est indépendante de notre contribution. Elle peut être remplacée par une autre technique de quantification si besoin. La mise en œuvre de cette solution sera discutée et présentée durant la présentation de notre travail.

3.2.4 Détection des changements sur l'arrière-plan

La détection des changements effectués sur l'arrière-plan est un sujet proche de ce qu'on cherche à résoudre (détection des limites des actions). La plupart des travaux traitant ce sujet (discuté dans (58)) effectuent une segmentation de l'arrière-plan comme étant la première étape, puis proposent leurs méthodes. Les auteurs de (54) appliquent des méthodes statistiques pour identifier les moments lors desquels des modifications majeures sont effectuées sur l'arrière-plan, en comparant l'arrière-plan de chaque trame avec l'arrière-plan déjà estimée. Une autre méthode proposée dans (49) surveille les objets du premier plan pour identifier les moments où l'un d'eux devient stationnaire et par suite devient partie intégrante de l'arrière-plan.

Là encore, nous ne cherchons pas à produire une segmentation de l'arrière-plan dans notre contribution, et de fait les limites des méthodes de segmentation citées n'ont pas de répercussion dans notre travail. Le comportement des pixels est étudié, qu'il corresponde à l'arrière-plan ou à un objet mobile, de la même manière.

3.3 Technique de quantification – Codebook

3.3.1 Définition

Dans le domaine du traitement automatique de contenus vidéo, la quantification des triplets RGB est une technique utilisée pour construire un modèle de l'arrière-plan. Ce type d'outils est largement utilisé dans les travaux de classification des pixels de premier et d'arrière-plan (en particulier pour la vidéo surveillance). L'utilisation principale consiste à construire un modèle identifiant les couleurs de l'arrière-plan, et par suite à soustraire ces informations de la scène à chaque instant pour ne conserver que l'information des pixels correspondant à des objets d'intérêt supposés composer le premier plan. Notre travail ne vise pas à segmenter les éléments qui composent spatialement une scène mais à détecter tout changement permanent de chaque pixel. Nous avons discuté dans la section précédente des méthodes les plus populaires pour construire un modèle d'arrière-plan par quantification (Mélange de gaussiennes et Codebook). Dans notre approche, le choix de la méthode de quantification n'est pas crucial. Nous avons sélectionné la méthode de quantification par Codebook en raison de sa simplicité, et de son aptitude à traiter des flux[1] dont les caractéristiques sont susceptibles d'évoluer au cours du temps. Son principe, relativement standard en quantification, consiste, pour une séquence de valeurs RGB prises par un pixel au cours du temps, à associer aux triplets les plus proches un niveau de quantification appelé Codeword. L'ensemble de tous les Codewords d'un pixel forme son Codebook.

[1] Dans le contexte de notre travail principal ("Synchronisation d'une vidéo avec le texte qui le décrit"), nous ne considérons actuellement que de courts enregistrements. Mais nous choisissons d'implémenter toutes nos méthodes de manière à ce qu'elles soient adaptées plus tard au traitement de longs flux.

Il existe de nombreuses variantes visant à identifier les groupes de triplets proches (ne serait-ce qu'en raison des multiples possibilités de définir la notion de "proximité" entre triplets), et de leur associer un Codeword. Nous avons choisi la méthode de KIM *et al.* (45) décrit dans le paragraphe suivant.

3.3.2 Méthode de KIM *et al.* (45)

Les valeurs RGB des pixels sont affectées d'une façon ou d'une autre par des facteurs externes comme le bruit, la variation de luminance, *etc...* Nous considèrerons que les variations induites par ces phénomènes sont de faible amplitude. Le premier objectif de la quantification est alors d'associer une seule et même couleur aux multiples triplets qui peuvent être associés à un pixel au cours du temps en raison de ces faibles variations.

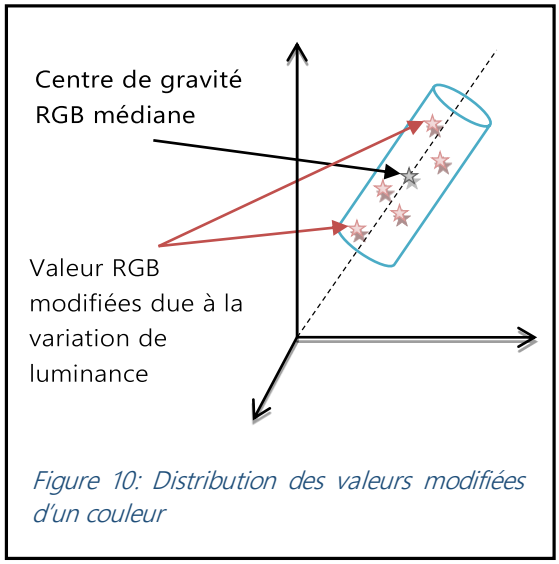


Figure 10: Distribution des valeurs modifiées d'un couleur

Mais comment savoir si deux valeurs RGB doivent être associées à un seul ou à deux niveaux de quantification différents ? À défaut de choisir un espace de représentation moins sensible aux changements d'éclairage, les auteurs ont étudié comment les valeurs RGB représentant un même point d'une scène varient sous l'effet d'un changement d'illumination. Selon leurs expérimentations, les triplets RGB obtenus sous différentes conditions d'éclairage standard se distribuent dans un sous-espace de forme allongée, étiré selon l'axe reliant le centre de gravité des triplets à l'origine du repère (Figure 10). Les auteurs ont choisi de représenter ce sous-espace par un cylindre. Ce cylindre est considéré comme étant un Codeword : l'ensemble des triplets RGB situés à l'intérieur sont associés à un même niveau de quantification.

Afin de construire un modèle d'arrière-plan, les auteurs construisent pour chaque pixel un Codebook formé par un ensemble de Codewords. La méthode requiert d'exprimer pour chaque Codeword les paramètres suivants :

R, **G** et **B**	les composantes moyennes des triplets RGB associés à ce Codeword. Ils correspondent au centre de gravité du cylindre.
$\mathbf{I}_{min}$, $\mathbf{I}_{max}$	l'intensité minimale et maximale associées aux valeurs RGB du Codeword. Ces valeurs identifient la dimension du cylindre. Des éléments de discussion sur les paramètres liés à la dimension du cylindre sont abordés plus loin.

F	le nombre de triplets associés au Codeword au cours du temps. C'est la fréquence de ce Codeword.
λ	le plus long intervalle durant lequel le Codeword n'est pas apparu.
p et **q**	la position de la première/dernière apparition du Codeword au cours du temps.

Cette méthode est proposée à l'origine pour construire et modifier un modèle d'arrière-plan en temps réel. À chaque nouvelle valeur, la procédure suivante est appliquée afin de mettre le modèle à jour :

→ Si la nouvelle valeur n'appartient à aucun cylindre déjà créé dans le Codebook, alors créer un nouveau Codeword tel que les coordonnées du centre de gravité du cylindre soit les composantes RGB de la nouvelle valeur.

→ S'il elle appartient à un cylindre, alors en modifier ses paramètres pour prendre en compte la nouvelle valeur (incrémenter F, recalculer le centre de gravité, etc...)

Durant la mise à jour du Codebook, des intersections entre les cylindres des Codewords peuvent avoir lieu. Cela est dû au redimensionnement et au déplacement du cylindre à chaque nouveau triplet RGB ajouté. Ce point peut poser problème, mais les auteurs ne le traitent pas dans leur travail. Dans ce qui suit, nous ne prétendons pas résoudre ce problème car il n'a que très peu d'impact sur les résultats (du moins dans nos expériences). Nous proposons néanmoins des modifications qui aident à minimiser les collisions entre les cylindres.

Pour l'instant, chaque Codebook dans le modèle collecte toutes les couleurs prises par le pixel correspondant, même ceux du premier plan. Comme les auteurs construisent un modèle d'arrière-plan, ils cherchent à éliminer les Codewords correspondants au premier plan. Par hypothèse, ces Codewords sont caractérisés par une grande valeur de **λ** indiquant que la couleur correspondante n'est pas apparue depuis longtemps. Une fois le Codebook filtré, l'appartenance d'un triplet RGB à un Codeword identifié, conduit à décider si le pixel contient une information d'arrière-plan ou non.

Cette méthode de quantification donne des résultats suffisamment satisfaisants pour être référencée dans nombreux autres travaux (plus de 1100 articles : (46), (47), (48)). Toutefois, ce qui nous intéresse dans le travail de KIM et *al.* (45) ce n'est pas la méthode de segmentation en profondeur de plans, mais plutôt la technique de quantification des valeurs RGB. Quelques modifications sont toutefois nécessaires pour qu'elle soit complétement adaptée à nos besoins.

3.3.3 Modifications proposées

Dans cette section, nous décrivons les modifications apportées à la méthode originale de construction de Codebook de KIM *et al.* (45), afin de l'adapter à nos objectifs propres.

Nous proposons les modifications suivantes:

3.3.3.1 Les composantes de Codeword

R, **G** et **B**	ces composantes doivent être conservées pour indiquer le centre de gravité du cylindre correspondant. Et par suite, elles représenteront la couleur correspondante de ce Codeword.
$\mathbf{I}_{min}$, $\mathbf{I}_{max}$	l'intensité minimale et maximale des valeurs RGB sont éliminées de la liste des paramètres. Dans notre approche, nous allons utiliser une dimension statique du cylindre pour tous les Codewords (voir la section 3.3.3.2).
F	le nombre des valeurs RGB associées à ce Codeword sera conservé sans modification.
λ	Le but d'utilisation de cette composante était d'identifier les Codewords correspondant au premier plan dans le but de les éliminer du modèle. Dans notre travail, on cherche à construire un modèle qui contient tous les Codewords représentant l'historique des couleurs prises par un pixel. De ce fait, nous n'avons plus besoin de ce paramètre.
p et **q**	Pour la même raison que précédemment, la position de la première/dernière apparition d'un Codeword n'est plus utile dans notre système.

3.3.3.2 La dimension de cylindre

Dans la méthode de KIM *et al.* (45), le Codebook de chaque pixel est mis à jour en temps réel à l'arrivée de chaque nouvelle trame. Cette mise à jour consiste à modifier un Codeword déjà créé ou à en créer un nouveau. Ainsi, si la nouvelle valeur RGB d'un pixel appartient à un cylindre déjà créé dans le système RGB, alors les paramètres de ce Codeword sont modifiés, notamment le centre de gravité et les dimensions du cylindre. Ces changements induisent un déplacement du cylindre dans l'espace RGB. Par conséquence, des intersections géométriques entre des cylindres dans le système RGB peuvent avoir lieu, ce qui induit une ambiguïté au moment de la quantification.

Comme on l'a dit, KIM *et al.* (45) ne proposent pas une solution pour éviter les intersections entre les cylindres, et nous ne présentons pas ici une solution complète

de ce problème. Mais nous proposons une modification qui vise à réduire les possibilités d'intersection.

Rappelons que les motivations premières de l'utilisation d'un cylindre dont l'axe principal passe par l'origine comme sous-espace associé à un Codeword sont que ce sous-espace est supposé permettre de collecter tous les triplets RGB d'un point d'une scène soumis à de petites variations de la luminance. En considérant des environnements types ou des corpus finis pour l'application de notre méthode (cuisine, bureau, *etc*...), cette variation peut être mesurée de manière expérimentale sur des zones statiques de l'image. Les dimensions des cylindres peuvent être fixées par la suite sous la forme de constantes définies a priori.

L'utilisation d'une dimension constante des cylindres réduit la possibilité d'avoir des intersections mais les déplacements des cylindres continuent à en induire. Durant l'évaluation, nous avons remarqué que la fréquence des intersections est faible et qu'elles peuvent être ignorées. Plus tard (section 3.3.4), nous indiquerons comment on va les traiter.

Les expérimentations suivantes ont pour objectif de déterminer ces dimensions sur quelques exemples-type.

Nous exprimons le domaine de variation des triplets RGB des pixels dans différentes vidéos sans mouvement et sans modification artificielle des conditions d'éclairage. Les enregistrements sont effectués en utilisant la même caméra installée sur un trépied. Nous utilisons plusieurs types de source d'illumination, chacune prise séparément, pour en connaître l'impact sur les dimensions des cylindres. Les sources d'illumination sont :

- → Un éclairage fluorescent,
- → Un éclairage incandescent,
- → Un éclairage par LED,
- → La lumière de soleil directe,
- → La lumière de soleil indirecte.

Figure 11: Exemple des scènes utilisées.

La procédure suivante est appliquée à chaque pixel de chaque vidéo prise séparément :

On considère $(R_i, G_i, B_i) = \{(R_{i,1}, G_{i,1}, B_{i,1}), (R_{i,2}, G_{i,2}, B_{i,2}), \ldots, (R_{i,n}, G_{i,n}, B_{i,n})\}$, les valeurs successives des composantes RGB prises par le pixel X_i dans l'enregistrement. Le centre de gravité Gr_i du cylindre est le moyenne de toutes les valeurs inclues dans ce cylindre (Figure 12).

$$Gr_i \text{ (moyenne}(R_i), \text{ moyenne}(G_i), \text{ moyenne}(B_i))$$

Désignons par "**O**" le triplet de coordonnées (0, 0, 0). Nous exprimons une mesure d'écart entre les valeurs des triplets et le centre de gravité sous la forme :

$$H_i = \{| \|Gr_i\ O\| - \|P_{i,j}\ O\| |\}$$

où **j** varie de 1 à **n**, et où $P_{i,j}$ est le point $(R_{i,j}, G_{i,j}, B_{i,j})$

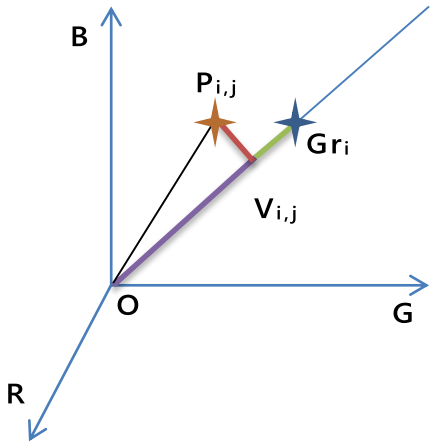
Ensuite, les distances entre les valeurs prises et l'axe allant de Gr_i vers l'origine "**O**" :

$$R_i = \{\|P_{i,j}\ V_{i,j}\|\}$$

Où :

→ $j=1 \rightarrow n$ et,

→ $V_{i,j}$ est la projection de $P_{i,j}$ sur l'axe $[O\ Gr_i]$



Gr_i : Moyenne des valeurs prises par le pixel X_i.

$P_{i,j}$: Une valeur prise par le pixel X_i à l'instant i.

Figure 12: méthode de calcul de la dimension du cylindre.

3.3.3.2.1 Résultats

Nous présentons des figures qui montrent les valeurs calculées pour chaque type de source d'illumination.

La Figure 13 montre les variations d'intensité des pixels. Pour chaque groupe de pixels représenté par l'axe horizontal, nous calculons leurs intensités maximale,

minimale et moyenne, ainsi que l'écart-type. Cela montre comment les triplets d'un pixel varient sans modification artificielle sur la scène.

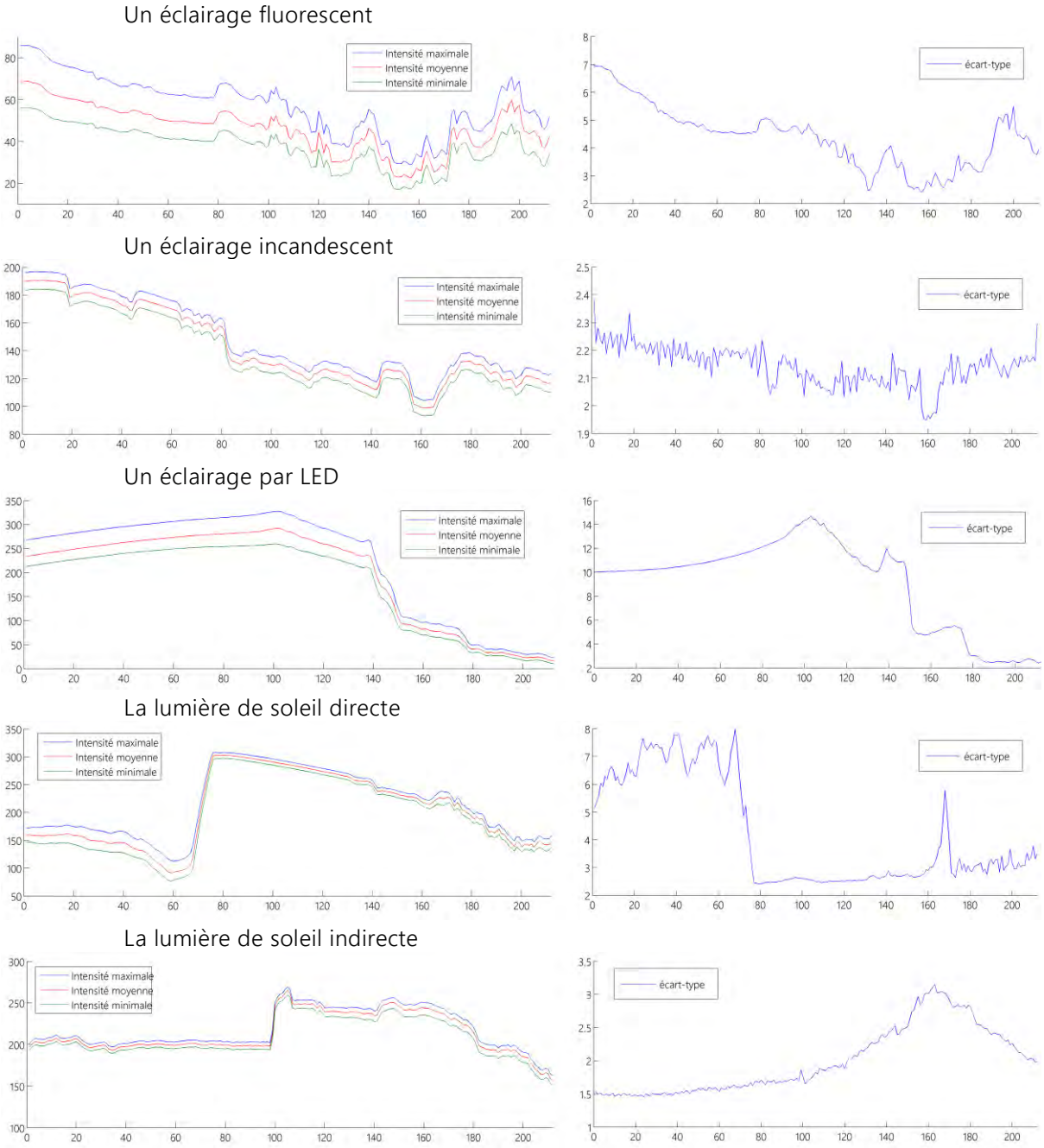


Figure 13: Les variations d'intensité sous différents éclairages.

Nous calculons maintenant les distances entre les triplets RGB d'un pixel et l'axe passant par l'origine et leur moyenne (les valeurs "PV" dans la Figure 12). Dans la Figure 14, nous indiquons pour chaque groupe de pixels (axe horizontal), la distance maximale, minimale et moyenne, ainsi que l'écart type des distances calculées.

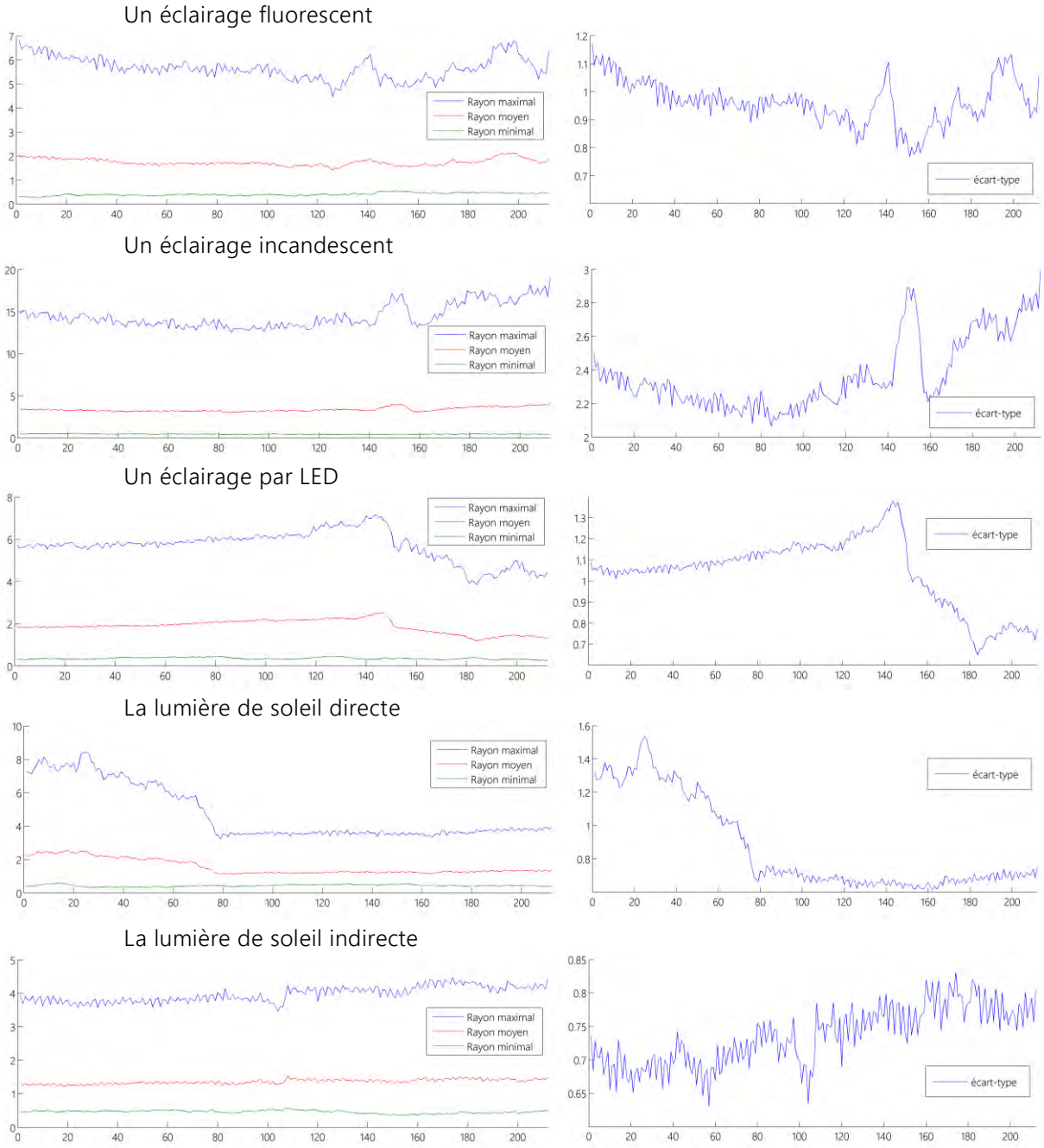


Figure 14: Les distances entre les triplets et l'axe allant du centre du repère vers leur moyenne ("PV" dans la Figure 12).

Nous remarquons que les variations d'intensité sous chaque source d'illumination varient de manière différente. Nous remarquons que les groupes de pixels varient également dans un intervalle différent sous un même éclairage car les pixels ayant une faible intensité (pixels moins éclairés : ombre, couleur foncée, *etc*...) varient moins que les pixels ayant une intensité élevée (pixels fortement éclairés : couleur claire).

Ces observations devraient nous conduire à fixer les dimensions des cylindres en fonction de l'intensité. Toutefois, nous remarquons que dans les pires cas, les positions des triplets RGB ne s'éloignent que modérément de la position moyenne. Pour des

raisons de simplification de calcul, nous choisissons de fixer arbitrairement les dimensions du cylindre en prenant l'écart-type des positions des triplets RGB autour de la position moyenne. Des expérimentations plus avancées seraient nécessaires pour trouver les dimensions optimales. Nous choisissons d'utiliser les dimensions suivantes :

1) *Hauteur du cylindre :*

$$\mathbf{GrH = écart\_type\ (H_i) \times 2}$$

2) *Rayon du cylindre :*

$$\mathbf{GrR = écart\_type\ (R_i)}$$

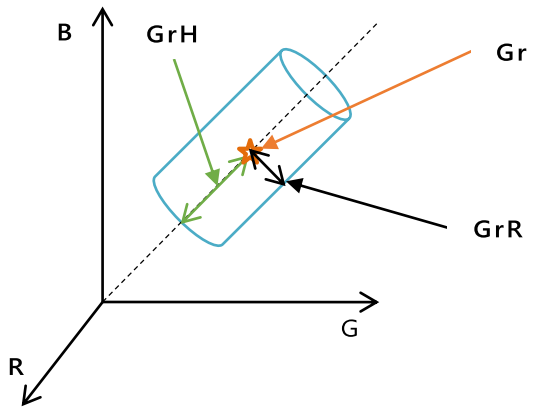


Figure 15: Dimension du cylindre du pixel i.

3.3.4 Construction et utilisation

Dans cette section, nous décrivons la démarche suivie pour construire les Codebooks, ainsi que la procédure pour la quantification des triplets sous forme de Codeword.

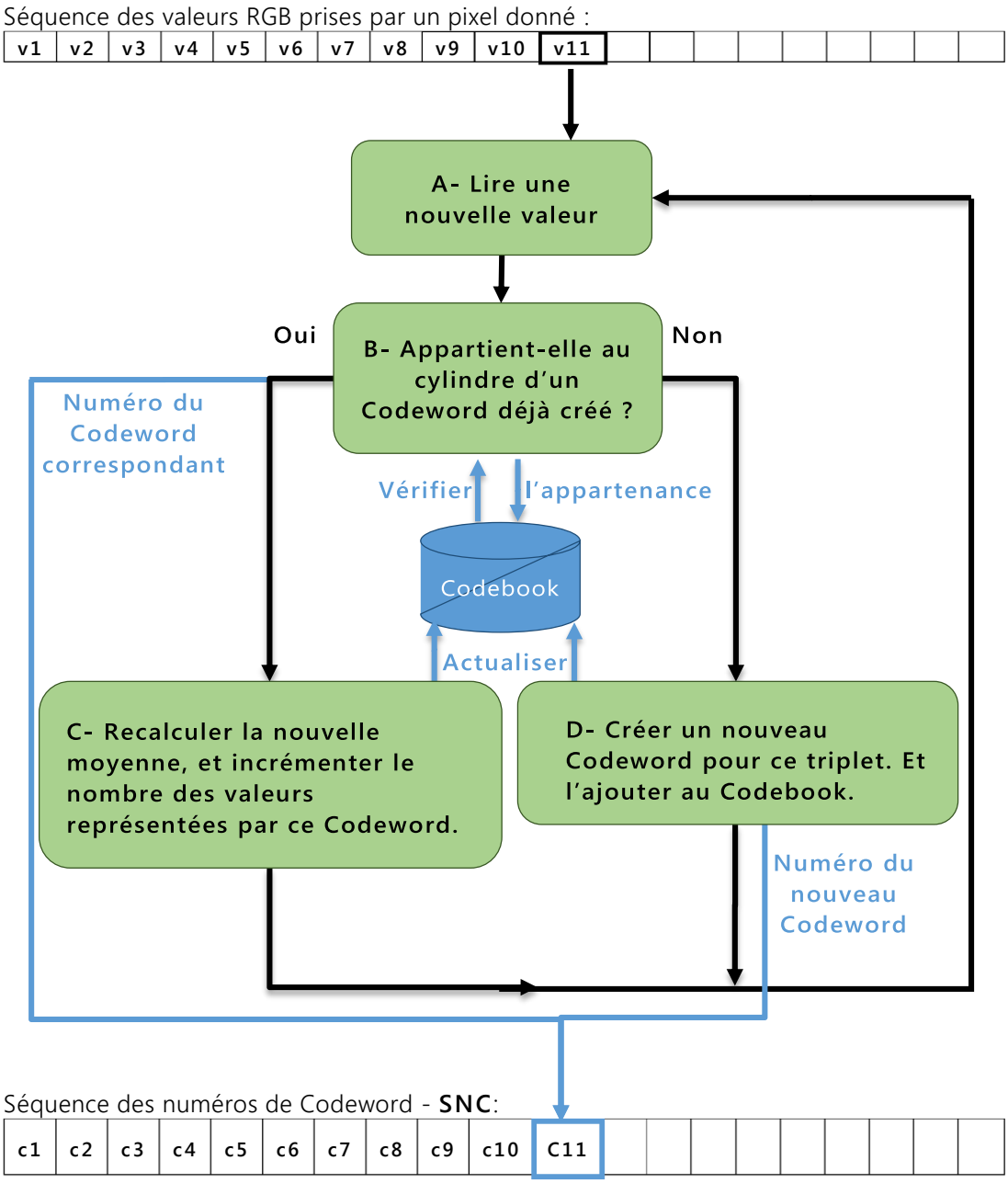


Figure 16: Procédure de construction du Codebook.

1) Lire une nouvelle valeur

Dans cette étape de la quantification, nous lisons un nouveau triplet RGB correspondant à un pixel donné comme s'il faisait partie d'un flux, ou plus exactement d'une séquence.

2) Appartient-elle au cylindre d'un Codeword déjà créé ?

Par la suite, nous désignerons par **SNC** la Séquence des Numéros de Codeword. La **SNC** décrit le chemin suivi par un pixel entre les Codewords dans l'espace RGB.

Dans un Codebook, chaque Codeword est associé à un identifiant qui est son numéro d'ordre de création. La quantification consiste à trouver le numéro de Codeword qui correspond au triplet RGB lu, en identifiant le cylindre qui contient ce triplet dans l'espace RGB. Le numéro de ce cylindre va représenter ce triplet dans la **SNC**.

Deux cas sont possibles :

a. Ce triplet appartient à un cylindre déjà créé

Si le triplet appartient à un cylindre, alors il lui sera associé, et par conséquence le numéro du Codeword est ajouté à la **SNC** du pixel.

Ensuite, nous modifions les paramètres de ce Codebook pour prendre en considération le triplet nouvellement associé. On recalcule le centre de gravité (moyenne des valeurs associées) et on incrémente le paramètre "**F**" dans le Codeword correspondant.

b. Ce triplet n'appartient à aucun cylindre

Dans ce cas, nous créons un nouveau Codeword dans le Codebook, de sorte que le centre de gravité de son cylindre soit le nouveau triplet et ses autres paramètres soient initialisés ("**F**", "**p**" et "**q**" – *cf.* 3.3.3.1). Le numéro de ce Codeword est alors ajouté à la **SNC**.

c. Ce triplet appartient à plusieurs cylindres

En actualisant les centres de gravité des cylindres à chaque nouvelle valeur ajoutée, des collisions sont possibles entre eux. Par conséquence, il est possible qu'une nouvelle valeur appartienne à deux cylindres (ou plus). Dans ce cas, nous choisissons d'associer la nouvelle valeur au Codeword le plus proche (en termes de distance avec le centre de gravité).

À la fin de cette procédure, la séquence de triplets est alors associée à la séquence des numéros des cylindres qui les contiennent.

3.4 Automate de décision

Le but est maintenant de proposer un outil capable d'analyser la **SNC** de chaque pixel afin de trouver les instants où ce pixel subit une modification entre deux états stables de couleur (appelée "**transition**").

En d'autres termes, on considère qu'un pixel subit une transition si on observe sur sa **SNC** un passage sans retour d'un Codeword à un autre. Nous considèrerons qu'une "**transition**" est symptomatique d'une modification persistante de l'environnement sur lequel agit l'opérateur dans la scène. En conséquence, les instants correspondant à ces passages indiquent des limites d'actions (en regard des restrictions que nous avons pu apporter sur la définition des actions) au niveau pixel. Notre but dans cette section est celui de la localisation de ces instants.

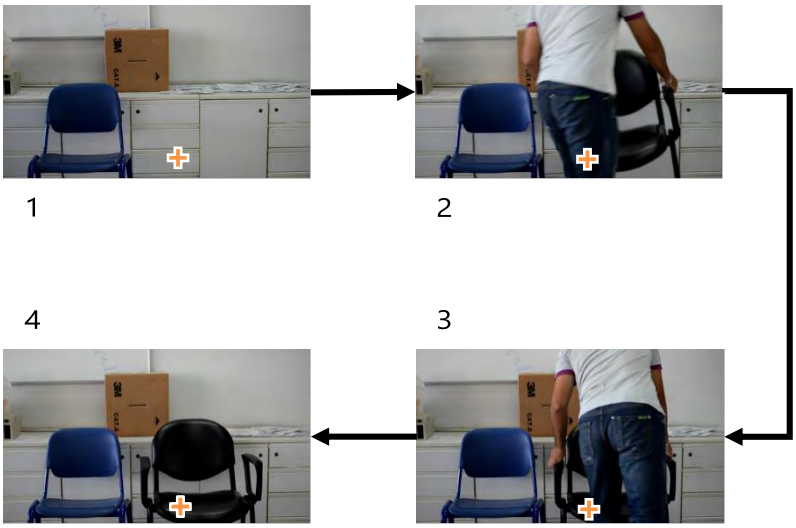


Figure 17: Exemple d'une action produisant une modification persistante de l'arrière-plan.

Cette figure montre une action où le symbole "+" représente la position d'un pixel subissant une transition entre deux couleurs persistantes de la couleur **blanche** vers la couleur **noire.**

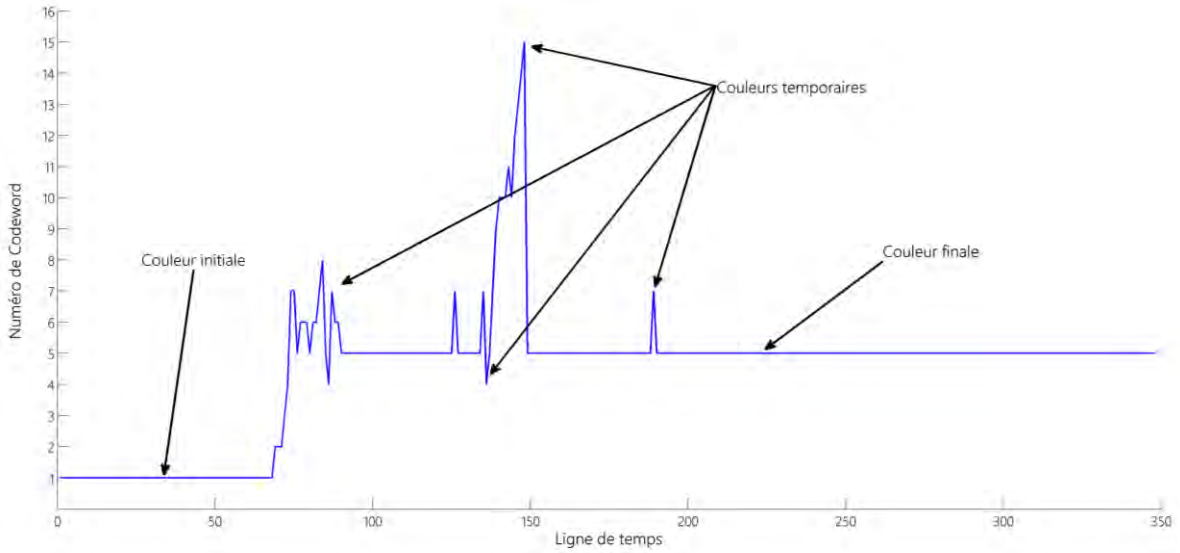


Figure 18 : SNC d'un pixel subissant une modification permanente (Figure 17). Cette figure présente la SNC de ce pixel tout au long de la vidéo. Où la couleur initiale/finale correspond à la couleur Blanche/Noire.

Pour identifier le type de problème à résoudre, considérons la **SNC** suivante :

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
1	1	1	1	1	1	1	1	1	2	2	2	3	3	3	3	3	4	4	4	4	4	4	4	4	4

Figure 19: exemple type de SNC.

Dans la **SNC** de ce pixel, il existe trois modifications : 9-10 (1→2), 12-13 (2→3) et 17-18 (3→4). Mais cet exemple appel à la détection d'une seule transition du Codeword "1" au Codeword "4" d'une manière indirecte. Nous pouvons considérer que les Codewords "2" et "3" correspondent au passage de l'opérateur dans le premier plan – à l'instar de l'exemple donné dans la Figure 17. Dans cet exemple, nous devons donc considérer que les changements de Codeword (9-10, 12-13 et 17-18) ne représentent pas des limites d'actions effectives. Cela pose le problème de trouver un critère permettant d'identifier les Codewords d'arrière-plan pour réduire la recherche des transitions en s'appuyant uniquement sur ces codes.

3.4.1 Identification des Codewords d'arrière-plan

Afin d'identifier les Codewords correspondants à l'arrière-plan, il faut trouver un critère qui les diffère des autres Codewords.

3.4.1.1 Codeword d'arrière-plan (état stable)

Les Codewords de l'arrière-plan sont caractérisés par la longueur de la répétition du code qui leur est associé dans la **SNC**. Ces durées sont proportionnellement plus longues que les autres. Nous parlerons dans ce cas d'état stable du pixel ou du Codeword associé. Mais la notion de "longueur" est subjective. Nous pourrions définir un nombre de répétition minimal "**L**" permettant d'identifier un état stable. Mais cette constante dépend de plusieurs facteurs (Vitesse d'exécution, type d'action, nombre d'objets en mouvement, *etc*...) qui ne permettent pas d'en préciser une valeur qui conviendrait pour le traitement de tous les cas. De ce fait, nous proposons l'utilisation de plusieurs valeurs de "**L**" séparément, pour localiser des transitions candidates. Les positions ainsi obtenues sont ensuite combinées après à l'aide d'un système dédié.

Dans la section suivante, nous considérons que "**L**" (appelé "**durée de stabilité**") a une seule valeur fixée a priori. La prise en considération de différentes valeurs de "**L**" sera abordée plus tard.

3.4.1.2 Codewords temporaires

Les Codewords qui correspondent au premier plan ne génèrent habituellement que des répétitions de courte durée dans la mesure où ils correspondent à des objets en mouvements. Nous parlerons alors d'état temporaire.

3.4.2 Localisation des transitions

Rappelons qu'une transition au niveau du pixel indique une modification persistante de couleur, ce qui signifie que ce pixel fait partie d'une modification au niveau de la scène. Dans notre cas, cette modification correspond à une limite d'action (début ou fin). Le but de ce qui suit est d'identifier des positions candidates des transitions à l'échelle du pixel. Nous verrons plus loin comment l'ensemble des

positions candidates de tous les pixels est pris en considération pour inférer des propositions à l'échelle de la trame.

En général, les Codewords correspondant à l'arrière-plan sont identifiés par une répétition d'au moins "**L**" symboles. Donc, l'étude du chemin suivi par chaque pixel entre les Codewords nous permet de localiser les transitions. Dans ce qui suit, nous considérons les chemins possibles dans une **SNC** en identifiant les Codewords de l'arrière-plan. La Figure 20 illustre comment une **SNC** peut être représentée de manière synthétique en mettant en exergue la comparaison de la longueur des répétitions avec le paramètre **L**.

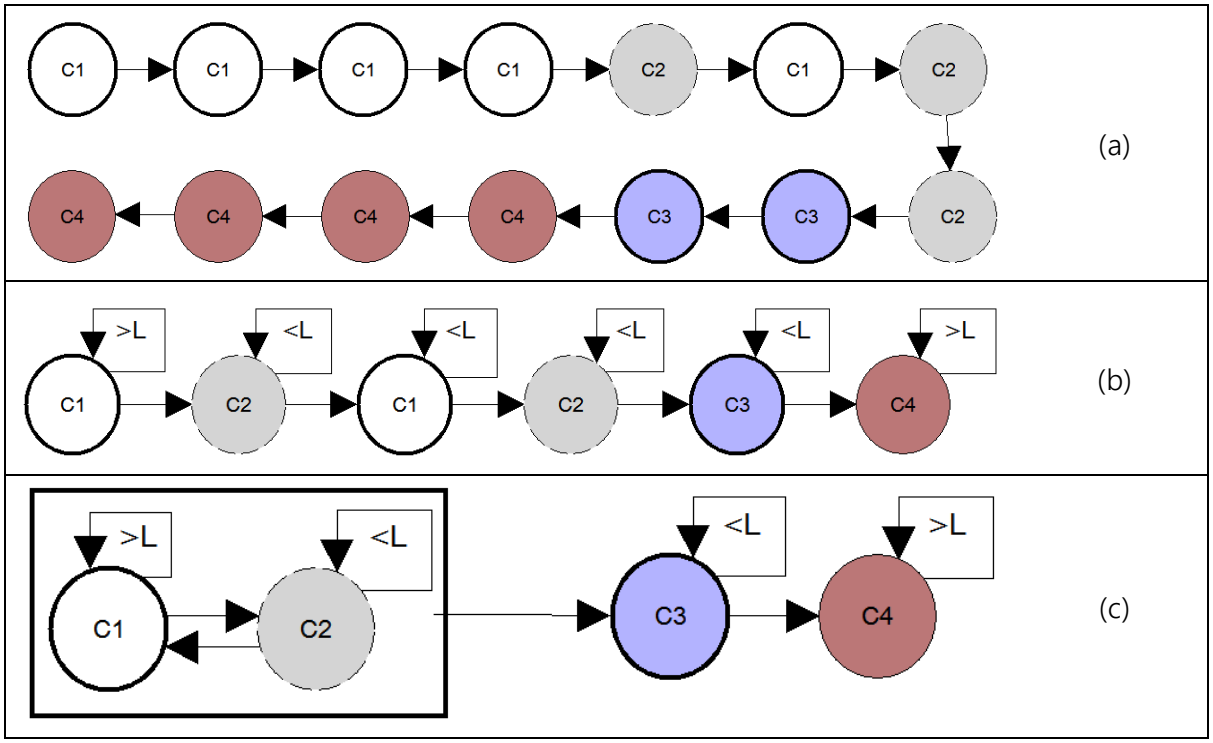
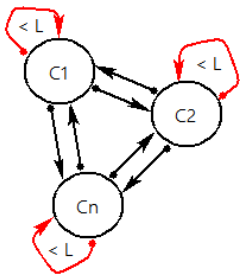


Figure 20: Représentation graphique d'une SNC. (a) La SNC d'un pixel formé de quatre Codewords différents ("C1", "C2", "C3" et "C4"). (b) Regroupement des répétitions successive du même Codeword, en indiquant si cette répétition est plus longue ou plus courte que "L". (c) Représentation finale de la SNC en (a). la notion "<L" (resp. ">L") signifie une répétition successive plus courte (resp. longue ou égale) que une valeur "L".

Cas I	
	Ce type de SNC correspond à un pixel pour lequel on n'est pas capable d'identifier un Codeword d'arrière-plan. Ce pixel ne subit aucune modification persistante dans sa SNC. Dans ce cas, notre système ignore ce pixel car on n'a aucune information concernant le type des Codewords.

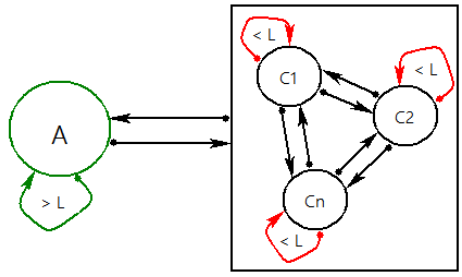
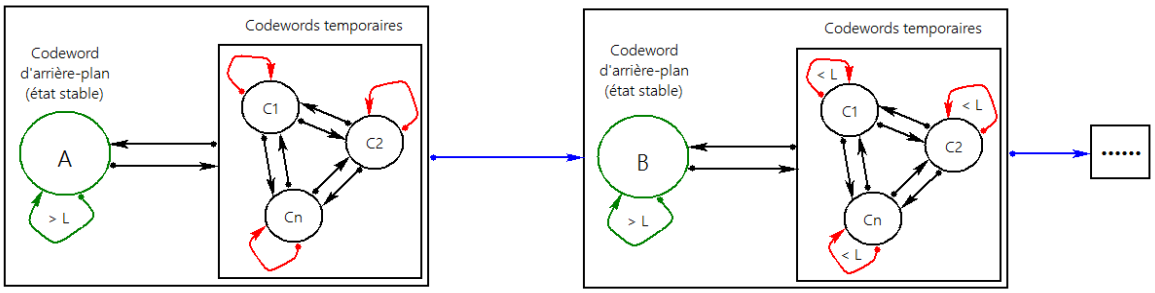
Cas II	
	Ce cas correspond à un pixel dont le Codeword d'arrière-plan est bien identifié. Mais durant la vidéo, ce pixel subit des modifications non persistantes dans sa SNC avant de revenir sur le même et unique codeword d'arrière-plan. Cela signifie que l'arrière-plan n'est jamais modifié à l'emplacement de ce pixel. Dans ce cas, il n'existe pas de transition à détecter. Les pixels ayant ce type de SNC ne représentent aucune modification durable de la scène. Ils sont donc ignorés durant la recherche des limites d'actions.
Cas III	
	
C'est le cas général d'un pixel qui subit des modifications persistantes dans sa **SNC**. La localisation de toutes les transitions dans cette **SNC** conduit à localiser toutes les limites d'action que ce pixel est susceptible de mettre en évidence. Dans ce cas, La première flèche entre les blocs rectangulaire indique une transition indirecte entre deux Codewords d'arrière-plan (ici, "**A**" et "**B**"). Dans ce cas, la transition indirecte correspond à une transition de "**A**" vers "**B**" en passant par des Codewords temporaires ("$\mathbf{C_i}$"). Nous ne sommes pas sûrs de la position exacte de la transition, mais nous savons qu'elle a lieu à un certain moment entre la dernière apparition du Codeword "**A**" et le début de la stabilité du Codeword "**B**". C'est le cas – par exemple – d'un opérateur qui effectue une action sans qu'elle soit directement visible par la caméra.	

Tableau 1 : Chemins possibles d'une SNC. On désigne par {C1, C2, …, Cn} des Codewords temporaires, et par "A" et "B" des Codewords d'arrière-plan.

Dans le cas III du Tableau 1, la première transition a lieu entre la dernière apparition du Codeword "**A**" et le début de la stabilité du Codeword "**B**", ce qui soulève la question à propos de la position exacte de la transition dans cet intervalle.

Rappelons que la position de la transition indique le moment qui correspond à une limite d'action (début ou fin) du point de vue du pixel. Dans le cas où cette transition correspond à un début d'action, "**A**" est le Codeword d'arrière-plan avant l'action et "**B**" est le nouveau Codeword d'arrière-plan après l'action. Un pixel correspondant à un objet avant son déplacement passe par une situation stable (l'affichage de l'objet), puis par une transition lors du mouvement, puis à nouveau par une situation stable (l'affichage de l'arrière-plan). De la même manière, le pixel situé à l'emplacement où l'objet déplacé fini par être déposé passe par une situation stable (l'affichage de l'arrière-plan, puis par une transition, puis par une nouvelle situation stable (l'affichage de l'objet). De ce fait, lors d'une action de cette nature, la zone de transition qui se trouve à la fin de la répétition d'un Codeword marque soit le début, soit la fin de cette action. Nous avons fait le choix de repérer la position correspondant à la fin de la répétition comme étant soit le début, soit la fin de l'action (à défaut de pouvoir préciser plus exactement la position où pourraient se trouver les limites réelles de l'action à l'intérieur de la transition).

Dans l'exemple suivant, nous localisons cherchons et localisons la position de la transition effectuée :

3.4.2.1 Exemple

Soit la **SNC** suivante qui correspond à un pixel contribuant à une modification persistante:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
1	1	1	1	1	1	1	1	1	1	2	2	1	2	3	3	3	3	4	4	2	2	4	4	4	4	4	4	4	4	4

En utilisant **L**=7, on remarque que les Codewords "1" et "4" sont des Codewords d'arrière-plan (répété consécutivement plus que "**L**" fois). D'où la représentation graphique de la **SNC** :

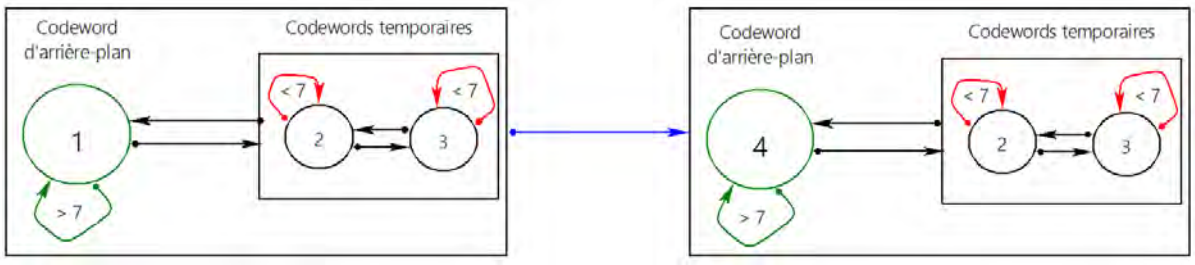


Figure 21: représentation graphique de la SNC. Elle correspond au cas III dans le Tableau 1.

Soit la représentation des positions où la limite de l'action (début ou fin) correspond au "**1**" et où les autres sont à "**0**" :

0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

3.4.3 Filtrage des **SNC**

Comme nous l'avons déjà mentionné, un Codeword est considéré stable s'il est répété sur une durée supérieure à "**L**" sans interruption. Mais ce cas peut être rare sur des données réelles potentiellement bruitées par d'autres phénomènes que des variations de luminance (dans la Figure 18 : on peut noter l'apparition d'un Codeword lié au bruit qui a interrompu la continuité de la répétition du deuxième Codeword stable). C'est pourquoi nous proposons de filtrer les **SNC** avant de commencer la recherche des limites d'actions.

3.4.3.1 Le choix du filtre

Il est important de mentionner que le but de ce filtrage n'est pas d'éliminer les Codewords correspondants au premier ou à l'arrière-plan, mais d'éliminer les Codewords liés au bruit. Pour distinguer cette catégorie, nous considèrerons que le bruit que nous cherchons à filtrer correspond à des valeurs aberrantes et ponctuelles susceptibles d'apparaître au milieu de la séquence. Nous formulerons donc l'hypothèse que ce bruit est de type "poivre et sel". Si tel est bien le cas, il peut être traité par un filtre médian (non-linéaire). L'application de ce filtre est effectuée via une fenêtre glissante, dont il reste à fixer la taille.

Toutefois, nous allons devoir adapter le filtre médian pour l'adapter à nos données. Un filtre médian est utile dans les cas où les valeurs de la séquence filtrée sont significatives, comme dans le cas d'une séquence d'intensités de pixels par exemple. L'ordonnancement, puis le choix de la valeur médiane s'effectue par rapport à toutes les valeurs dans la fenêtre. Dans notre cas, la **SNC** contient des numéros de Codewords indépendants et non-comparables en termes d'intensité (le numéro de Codeword correspond à son ordre d'apparition dans la vidéo). Par suite, le but n'est pas choisir le Codeword ayant un numéro médian. Nous choisissons d'utiliser la valeur la plus fréquente dans la fenêtre. De cette manière, un Codeword aberrant est remplacé par le Codeword le plus fréquent dans son voisinage (fenêtre).

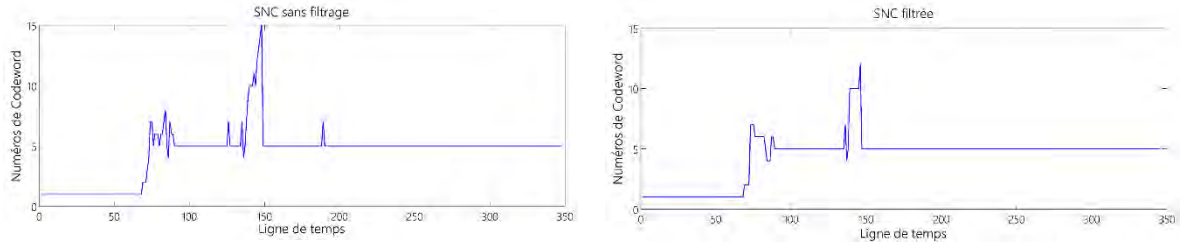


Figure 22 : Exemple d'une SNC avant et après le filtrage. On remarque l'absence des valeurs aberrantes dans la SNC filtrée. La taille de la fenêtre glissante est 3.

3.4.3.2 Taille de la fenêtre

Rappelons que notre objectif est de trouver dans une **SNC** des segments de longueur supérieure ou égale à un "**L**" (déjà défini – *cf.* 3.5.2) qui ne contiennent qu'un seul Codeword. Par suite, le choix d'une fenêtre de taille "**L**" apparaît comme

raisonnable pour maintenir la longueur des séquences de répétition de code. Ce filtrage n'est bien sûr possible que tant que le nombre de valeurs aberrantes reste petit.

3.5 Système de vote

3.5.1 Intégration des votes des pixels

À l'aide de la méthode décrite jusqu'à présent, nous caractérisons les transitions ayant lieu entre deux états stables d'un pixel (pour une valeur de "**L"** donnée). Du point de vue de ce pixel, nous considérons que l'existence d'une transition à un instant donné est susceptible de correspondre à une limite d'action dans la vidéo. Dans cette section, nous proposons un système visant à combiner les résultats obtenus sur tous les pixels sur l'ensemble des positions temporelles étudiées pour former une décision globale au niveau de la vidéo.

La position d'une transition, dans la **SNC**, entre deux Codewords stables indique la position probable d'une limite d'action. Mais, cette transition est généralement indirecte dans la mesure où des Codewords intermédiaires séparent le début de la fin de l'action (voir le cas III dans Tableau 1). Or, le plus souvent, les positions temporelles exactes du début et de la fin de l'action se trouvent au milieu de ces Codewords intermédiaires. Ce cas est présenté dans l'exemple 3.4.2.1, où les positions 14 à 18 séparent le Codeword 1 du Codeword 4. Les interventions sur la scène lors de cette phase sont cachées par l'opérateur (Figure 17 : image 2 et 3). Par contre, nous sommes sûrs qu'une modification persistante est effectuée sur l'arrière-plan après la trame 13 (le Codeword stable "1" n'apparait plus avant le nouveau Codeword stable "4"). Comme on l'a mentionné avant, nous choisissons de placer la transition à la dernière apparition du premier Codeword stable "1" (trame 13).

De point de vue de ce pixel, il existe une limite d'une action à la trame numéro 13. La séquence est transformée en "**Vecteur des votes**" identifiant la position des transitions observées sur un pixel.

0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 23: Vecteur de votes.

Afin d'obtenir une décision globale au niveau de la vidéo, il suffit de calculer la somme des vecteur des votes de tous les pixels pour produire le **"Vecteur global des Votes"**. Dans ce vecteur, une valeur élevée informe sur la grande probabilité que la trame correspondante forme une limite d'une action (début ou fin).

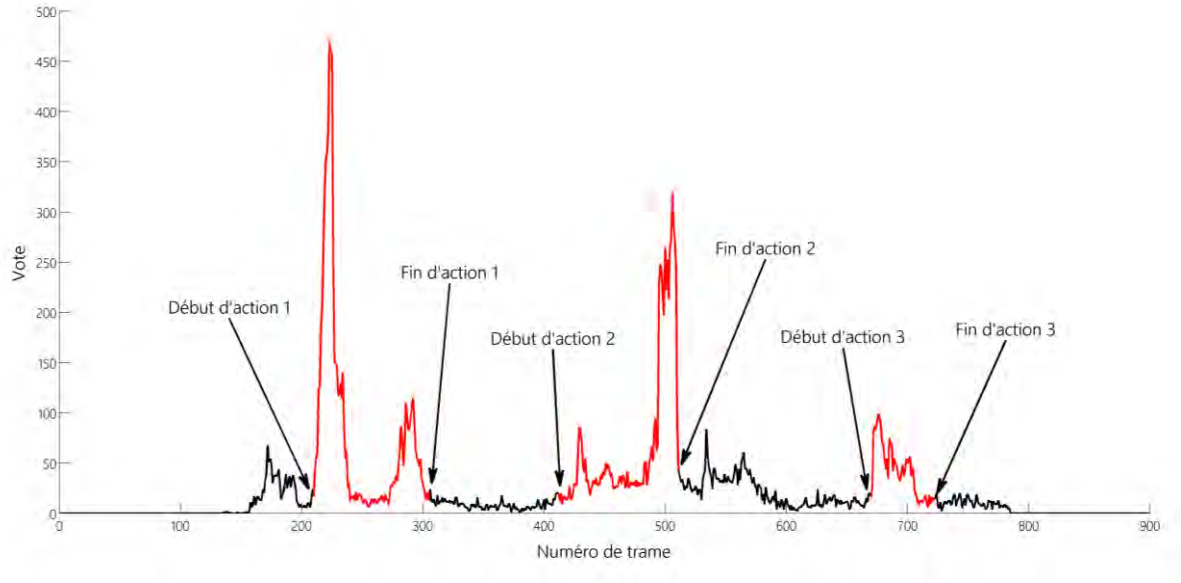


Figure 24: Exemple d'un "Vecteur global des Votes" d'une video à trois actions. Les flèche indiquent les débuts et les fins des actions selon la vérité terrain.

Même en ayant restreint fortement notre champ d'étude aux seules actions qui produisent une modification persistante de l'environnement de l'utilisateur, celles-ci ne se traduisent visuellement pas toutes de la même manière, et par suite elles ne produisent pas le même résultat sur le vecteur global des votes comme l'illustre la Figure 20. Dans le but de rehausser le contraste entre les valeurs correspondant à des transitions et les autres, nous pouvons considérer que la durée des états stables situés de part et d'autre de la transition, sont des indicateurs de confiance. Ainsi, plus ces états stables sont longs, plus il est probable que le pixel concerné affiche un objet immobile dans l'environnement. Nous proposons d'intégrer cette hypothèse dans notre système sous la forme d'une pondération des votes.

Nous utilisons pour cela comme poids la durée minimale parmi les durées des deux Codewords stables situés de part et d'autre d'une transition. De cette manière, une transition entre deux Codewords stables de courte durée vote avec une faible valeur et inversement.

$$VGV_L(t) = \sum_x \sum_y VVP(x, y, t)$$

Équation 1

où **VVP** est le vecteur de vote pondéré du pixel **(x, y)**. **VGVL** est le vecteur global des votes obtenu en fonction du paramètre **"L"**

La pondération des vecteurs des votes en utilisant la durée minimale pose un problème pour les limites d'action situées aux extrémités de la vidéo (au début ou à la fin de la vidéo). Par exemple, une action qui commence juste au début de la vidéo est associée à un vote qui est petit, car le premier Codeword stable (couleur initiale) n'apparaît pas longtemps avant qu'il ne soit changé. Ceci rend la détection de cette

limite plus difficile. Nous ne traiterons pas ce problème dans notre travail en considérant que si une attention particulière devait être portée aux actions qui pourraient apparaître en début ou en fin d'enregistrement, nous pourrions aisément définir une pondération adaptée en fonction de la position possible de la limite dans la vidéo et de la valeur du paramètre L.

Exemple :

Soit la **SNC** suivante :

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
1	1	1	1	1	1	1	1	1	1	2	2	1	2	3	3	3	3	4	4	2	2	2	4	4	4	4	4	4	4	4

La **SNC** des Codewords stables est :

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
1	1	1	1	1	1	1	1	1	1	-	-	1	-	-	-	-	-	4	4	-	-	-	4	4	4	4	4	4	4	4

Le vote reçoit pour poids le minimum (durée des "1", durée des "4") = minimum (10, 8)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

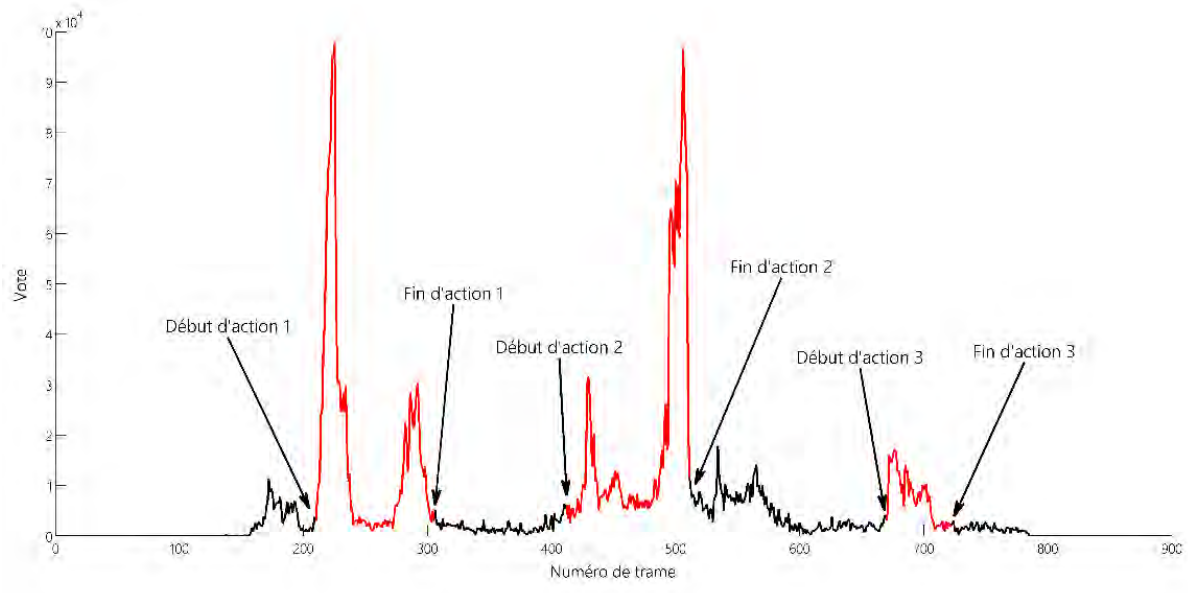


Figure 25: Vecteur de vote pondéré de la même vidéo que celle de la Figure 24.

La Figure 25 reprend la courbe de la Figure 24 après pondération des votes. Les parties en rouges de la courbe représentent les positions des actions dans la vidéo issues de la vérité terrain. Les limites des actions sont clairement représentées par des grands pics aux positions convenables (au moins pour les actions : 1 et 2). L'extraction de ces pics doit nous permettre d'identifier les limites des actions.

Pour montrer l'impact de la pondération, nous calculons la mesure suivante qui souligne les contrastes des pics correspondants aux actions, par rapport à la moyenne de la courbe.

Pour chaque vidéo "**v**", nous calculons :

$$M_v = \underset{p \in P}{\text{moyenne}} \left(\frac{VGV_v(p)}{moyenne(VGV_v)} \right)$$

Équation 2

Où :

- → $\mathbf{VGV_v}$ est le vecteur global de votes de la vidéo **"v",**
- → **P** est l'ensemble des indices des pics dans $\mathbf{VGV_v}$,
- → **p** est l'indice du pic qui correspond à une limite d'action.

Ensuite, nous calculons la mesure suivante de contraste:

$$Contraste\_moyen = \underset{v \in V}{\text{moyenne}}(M_v)$$

Équation 3

Les valeurs calculées de cette mesure en utilisant le vecteur global de votes pondérés et celui sans pondération, sont :

	Sans pondération	**Avec pondération**
Contraste moyen	2.2052	2.7904

Tableau 2 : La mesure de contraste du vecteur global de vote sans/avec pondération.

Nous remarquons que la pondération a augmenté l'amplitude des pics qui correspondent à des limites d'action par rapport aux autres valeurs, ce qui les rend plus aisément détectables.

3.5.2 Durée minimum d'occupation d'un état stable - L

Rappelons qu'un Codeword est considéré stable s'il existe un segment – dans la **SNC** – formé uniquement de ce Codeword, et de longueur supérieure ou égale à une valeur "**L"**. En se basant sur cette valeur, nous avons proposé un automate et un système de vote pour localiser les limites des actions. Mais il est difficile de trouver une valeur de "**L"** qui soit adaptée à toutes les situations car elle dépend de plusieurs facteurs (Type d'action, vitesse d'exécution, *etc*...) qui ne sont pas les mêmes dans tous les enregistrements.

Pour garantir une certaine indépendance de la méthode avec la valeur de ce paramètre, nous proposons de définir un intervalle des valeurs possibles de "**L"**, et de calculer le vecteur global des votes pour chacune des valeurs de cet intervalle. Dans la suite de ce chapitre, nous définissons les limites de cet intervalle de la manière suivante :

→ Bord inférieur = 3
c'est la durée minimum pour qu'un Codeword soit considéré comme stable,

→ Bord supérieur = longueur du plus grand segment d'une répétition d'un même Codeword dans une **SNC** où plusieurs Codewords apparaissent.

La détection des limites des actions pourra alors s'effectuer sur le résultat du calcul de la somme des vecteurs globaux des votes produits pour chaque valeur de "**L**". Nous appellerons ce nouveau vecteur, le vecteur final des votes :

$$VFV(t) = \sum_{L} \sum_{x} \sum_{y} VVP(x, y, t)$$

Équation 4

où **VVP** est le vecteur de vote pondéré du pixel **(x, y)**. **VFV** est le vecteur final des votes.

Exemple :

Soit la **SNC** suivante :

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
	1	1	1	1	1	1	1	1	1	1	2	2	1	2	3	3	3	3	4	4	2	2	2	4	4	4	4	4	4	4	4

Les vecteurs globaux de vote pour chaque "**L**" sont (on désigne par "**CS**" la **SNC** des Codewords stable et par **VVP** le vecteur de votes pondérés) :

L=3

CS	1	1	1	1	1	1	1	1	1	1	-	-	1	-	3	3	3	3	-	-	2	2	2	4	4	4	4	4	4	4	4
VVP	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	3	0	0	0	0	3	0	0	0	0	0	0	0	0

L=4

CS	1	1	1	1	1	1	1	1	1	1	-	-	1	-	3	3	3	3	-	-	-	-	-	4	4	4	4	4	4	4	4
VVP	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0

L=5

CS	1	1	1	1	1	1	1	1	1	1	-	-	1	-	-	-	-	-	-	-	-	-	-	4	4	4	4	4	4	4	4
VVP	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

L=6, 7, 8

VVP	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
VVP	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
VVP	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

VFV	0	0	0	0	0	0	0	0	0	0	0	0	40	0	0	0	0	7	0	0	0	0	3	0	0	0	0	0	0	0	0

Dans cet exemple, nous pouvons voir que pour rendre la méthode indépendante de la valeur de "**L**", nous introduisons du bruit dans le signal en sortie aux positions 18 et 23. Le vecteur final des votes indique toutefois un vote élevé à la position 13 qui reste suffisamment contrasté avec les autres valeurs pour permettre de l'identifier comme limite d'action.

De même, nous calculons la mesure de contraste (selon la définition donnée par l'Équation 3) pour montrer que l'utilisation de plusieurs valeurs de "**L**" améliore le contraste des pics qui correspondent à des limites d'action.

	Valeur unique de "L" **VGV**	**Valeurs multiples de "L"** **VFV**
Contraste moyen	2.7904	3.0134

Tableau 3 : mesure de contraste du vecteur global/final des votes.

3.5.3 Algorithme final

Pour chaque pixel p
Début
    Représenter la séquence des triplets RGB prises par p en termes de numéro de Codeword, à l'aide du Codebook (section 3.3.4);
Fin
Pour chaque durée L >= 3
Début
    Calculer le vecteur de vote de chaque pixel ;
    Déduire le vecteur global de vote ;
Fin
Calculer le vecteur final des votes = somme des vecteurs globaux

3.5.4 Détection des limites

L'objectif maintenant est d'identifier la position des votes élevés aux positions correspondant aux limites d'actions. La manière d'atteindre cet objectif est la dernière étape de notre méthode.

Par principe, les votes élevés associés à des positions de transitions correspondent à des maximums locaux. Toutefois, en raison de la taille et du contraste variable de la région avant et après modification dans la scène, il n'est pas possible de déterminer un seuil absolu qui permettrait d'extraire toutes les limites. La première partie de la méthode va consister à filtrer les maximums locaux candidats.

Dans notre cas, nous cherchons les positions qui correspondent à la valeur la plus grande par rapport à son voisinage. Pour cela, nous proposons une technique de filtrage pour contraster ce type de maximums dans le vecteur final des votes. La technique est basée sur la détection des maximums dans une fenêtre glissante.

Pour une taille donnée **T** de fenêtre, on initialise un vecteur **VT** à 0, puis on calcule :

$$V_T\,(i+p-1) = \frac{max(VFV\,(i:i+T))}{T}; \qquad i = 1 \rightarrow Fin - T$$

Équation 5

Où :

→ VGV est le vote final des votes, i est la position du début de la fenêtre glissante sur VFV, p est l'indice du maximum dans la fenêtre commençant à la position i.

La taille de la fenêtre est fortement liée aux types des actions dans la vidéo, car cette technique favorise un maximum unique dans un intervalle égal à la taille de fenêtre "**T**". Plus les limites des actions à détecter sont éloignées; plus la taille de fenêtre est longue. Dans le cas où cette information est indisponible, on propose d'utiliser plusieurs valeurs de "**T**", et puis combiner les résultats. Le calcul de moyenne de ces vecteurs résultants apparaît une manière raisonnable pour les combiner en un seul :

$$\mathrm{V\,(k)} = \underset{T}{\mathrm{moyenne}}(\mathrm{V_T(k)})\,; \qquad \mathrm{T = t\_min \rightarrow t\_max}$$

Équation 6

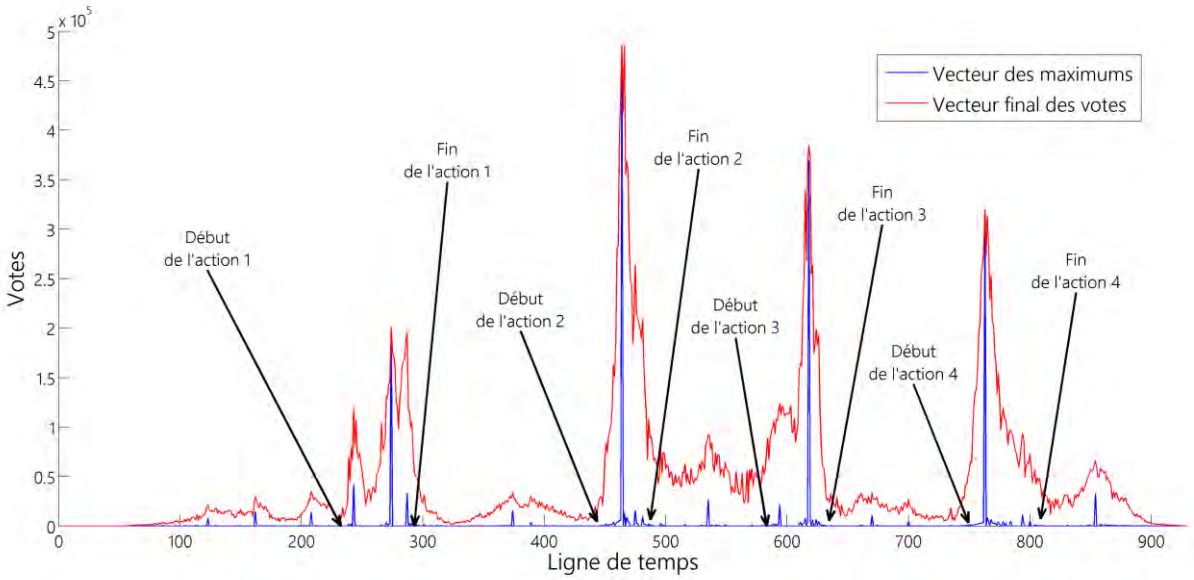


Figure 26 : le vecteur final des votes (en rouge), le vecteur des maximums (en bleu) et les positions réelles des limites. Ces courbes correspondent à une vidéo formée de quatre actions visant à déplacer quatre objets dans la scène.

Nous remarquons le contraste des pics dans la courbe des maximums (courbe bleu dans la Figure 26), ce qui les rend facilement détectables. De plus, nous remarquons que les trames ayant des votes élevés mais situées à proximité d'un grand pic, sont ignorées (ou ramenées à des valeurs négligeables) dans le vecteur des maximums (par exemple entre la trame 450 et 500).

La Figure 26 montre l'intérêt du vecteur des maximums pour localiser des maximums locaux pertinents dans la courbe finale des votes.

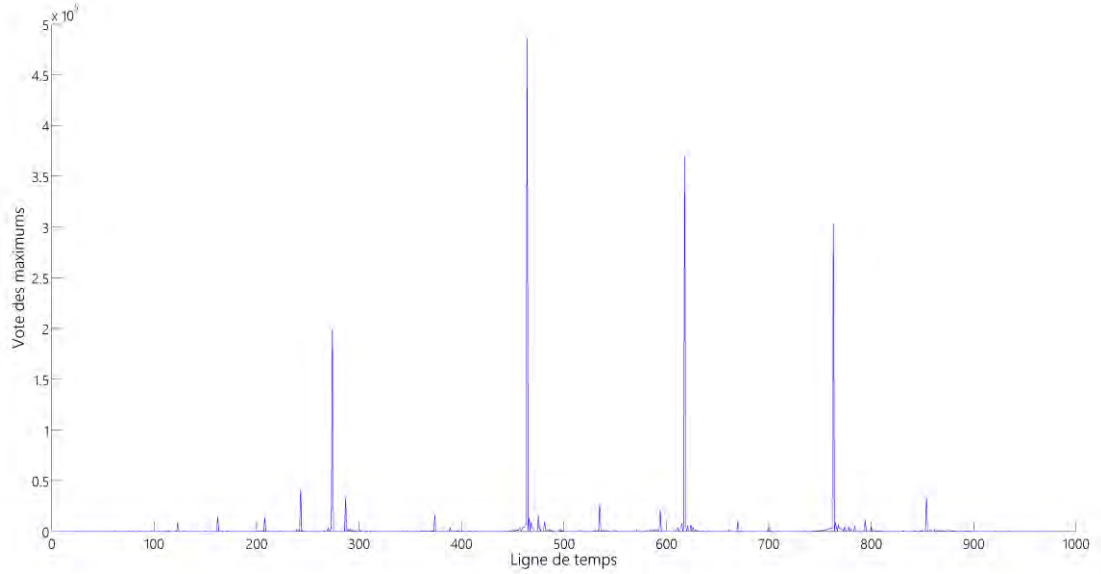


Figure 27: vecteur des maximums.

À partir de ce moment, les limites des actions dans la vidéo peuvent être détectées en identifiant les pics les plus élevés dans le vecteur des maximums.

Nous montrerons dans les sections suivantes que plus les pics sont grands, plus ils sont pertinents pour marquer une limite d'action. Cet outil est conçu pour être une partie du système globale de synchronisation. Dans ce système, le nombre de limites d'action est supposé connu a priori (à partir de l'analyse de texte). À partir de là, si **N** limites sont requises pour effectuer la synchronisation, nous n'aurons qu'à extraire les N plus grands pics. La question de fixer un seuil pour déterminer si un pic doit être retenu ou non est ainsi écartée.

3.6 Résultats et évaluation

Nous cherchons maintenant à évaluer la performance de notre système. Nous allons considérer pour cela un corpus composé de vidéos présentant des caractéristiques communes :

- → Les vidéos sont complètement enregistrées en utilisant une caméra stationnaire,
- → Les contenus des vidéos comportent des actions exercées par un opérateur dans son environnement et, en partie au moins sur son environnement,
- → Aucune modification majeure de l'arrière-plan n'est observable en dehors de celles effectuées par l'opérateur.

Nous n'avons pas trouvé de corpus publié proposant des vidéos susceptibles de servir de support à ces évaluations. Certaines auraient pu être utilisées indirectement

moyennant le coût d'une annotation adaptée. Il nous a semblé plus pertinent de produire un corpus adapté, contenant les éléments sensibles permettant d'illustrer les propriétés de nos propositions.

3.6.1 Le Corpus

Deux types des vidéos sont testés :

3.6.1.1 Contenus de synthèse

Nous avons choisi de construire des animations simples pour travailler sur des contenus contrôlés. Le but est de pouvoir éliminer d'une part les facteurs externes (bruit, mouvement de caméra, mouvements indésirables sur l'arrière-plan) qui agissent sur la performance du système, et d'autre part d'étudier le comportement et la sortie du système en regard de différents paramètres mis en œuvre pour la production des animations (dimension de l'objet soumis à l'action, vitesse d'exécution de l'action). Ces vidéos ont été créées sous Matlab et simulent des actions de déplacement de deux objets : une fois par un opérateur (Figure 28) et une autre sans opérateur (Figure 29).

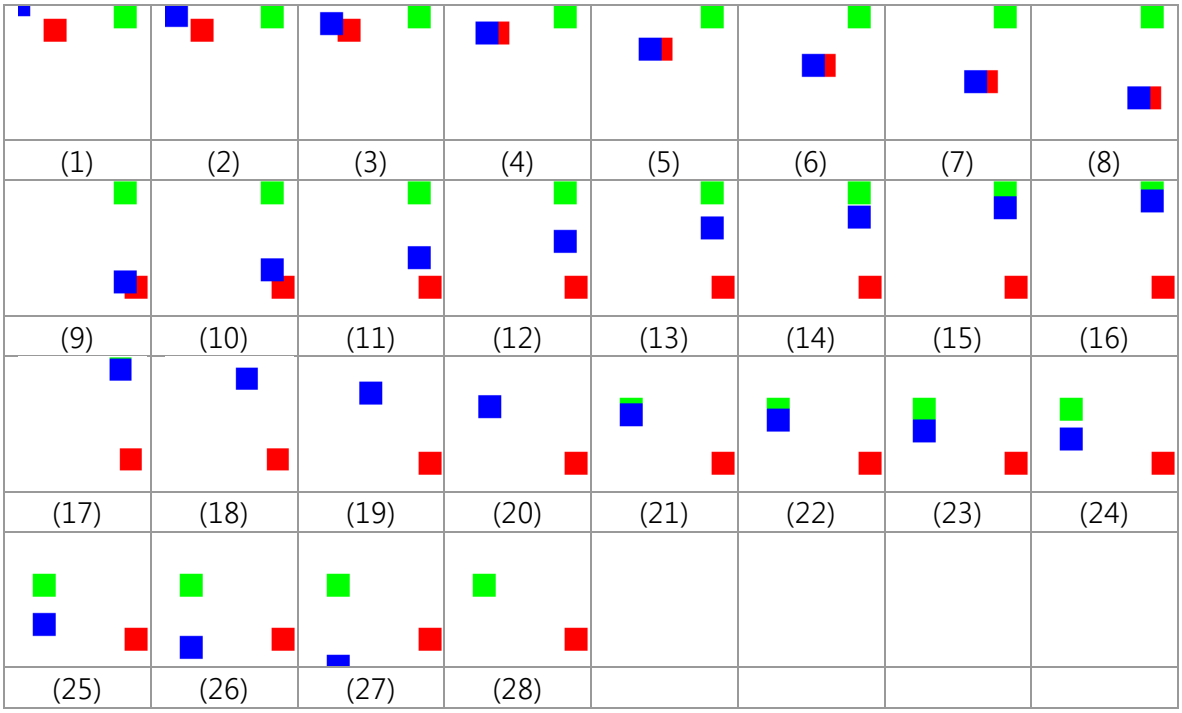


Figure 28 : "Vidéo 1". Un opérateur (carré bleu) déplace consécutivement deux objets (carrés rouge et vert) d'une position initiale à une position finale avant de quitter la scène. Cette vidéo formée de 260 trames de dimension 100x100, contient les exécutions de deux actions consécutives. Les dimensions des carrés est 17x17 pixels.

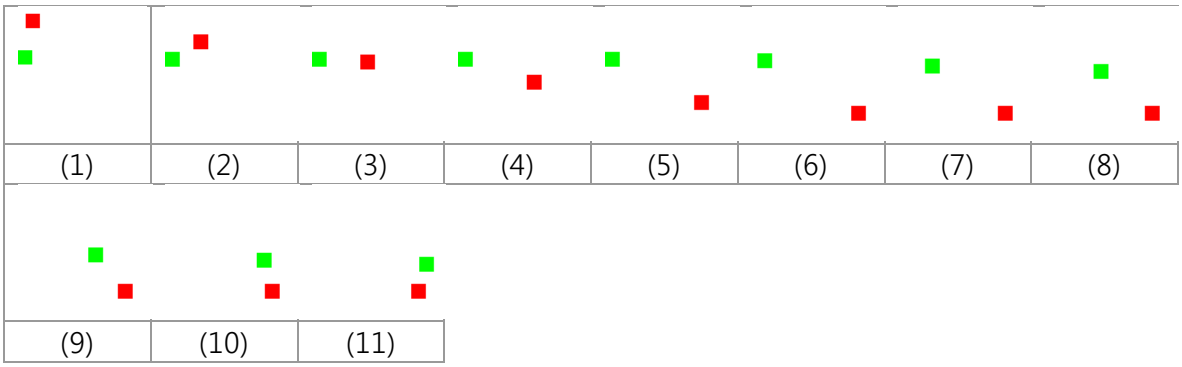


Figure 29 : "Vidéo 2". Deux objets changent de place consécutivement. Cette vidéo formée de 160 trames de dimension 100x100, contient deux actions (changement de place) sans opérateur. Les dimensions des carrés est 11x11 pixels.

Ces animations permettent de valider le système en vérifiant qu'il est applicable pour le moins sur des vidéos parfaites et ad hoc. Les résultats obtenus permettent d'observer le comportement des votes et de vérifier qu'il est conforme à nos attentes, à savoir qu'il présente de grandes valeurs de maximums locaux aux positions des limites.

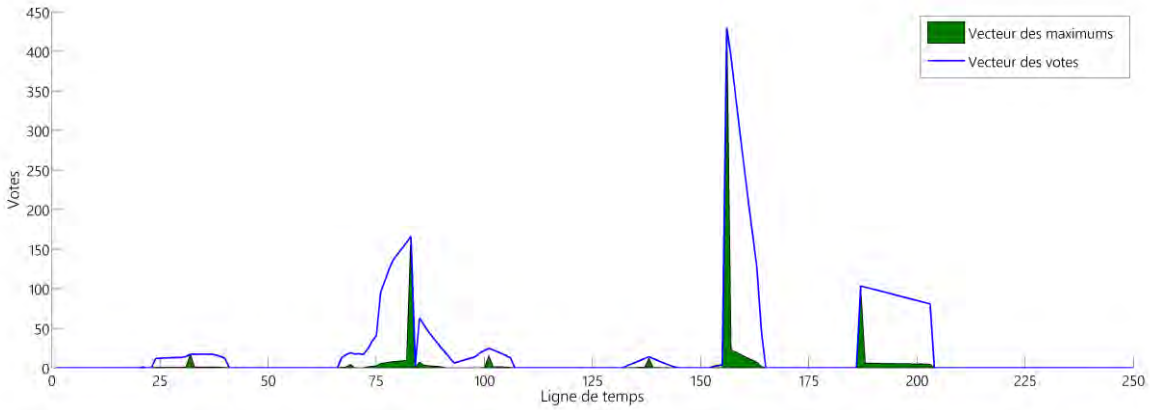


Figure 30: Les vecteurs de votes et des maximums de la vidéo 1 (Figure 28). On peut clairement identifier trois pics majeurs qui correspondent à trois des limites de la vérité terrain ("VT"). Notons qu'ici, la première limite n'est pas détectée car l'objet déplacé n'était pas stationnaire sur une période suffisamment longue pour générer un vote d'ampleur comparable aux suivants. Limites "VT" : 24, 83, 156 et 187. Résultats : 83, 156 et 187.

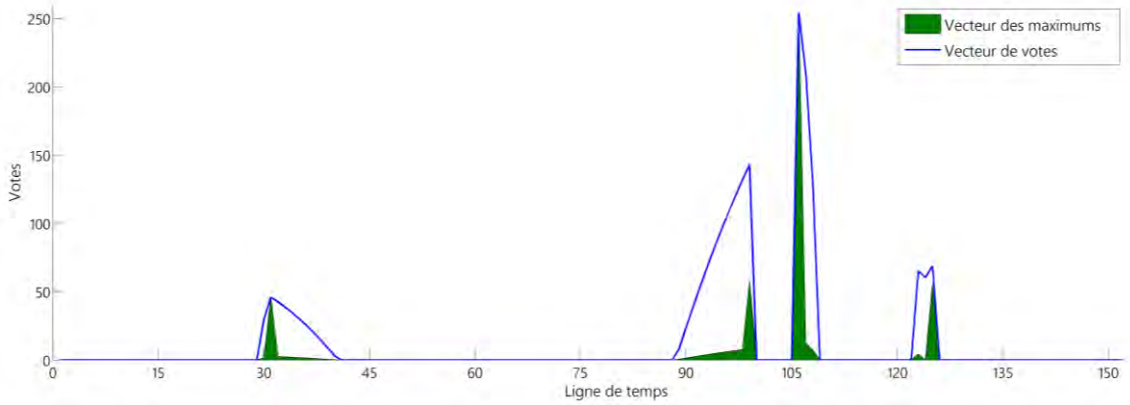


Figure 31: Les vecteurs de votes et des maximums de la vidéo 2 (Figure 29). Les quatre pics majeurs correspondent aux quatre limites "VT" des deux actions exercées sans opérateur. Limites "VT" : 31, 99, 106 et 125. Résultats : 31, 99, 106 et 125.

Selon les exemples ci-dessus, notre système associe des votes élevés aux points qui correspondent aux limites des actions. Dans ces figures, nous vérifions que nous obtenons bien les résultats attendus (grands pics aux limites d'actions) avant d'évaluer la méthode sur des données réelles.

3.6.1.2 Vidéos réelles

Notre corpus est formé de 16 vidéos ayant une durée totale égale à 693 secondes (20112 trames) et un nombre d'actions égal à 109 actions, ce qui produit 218 limites d'action à détecter (voir le Tableau 4). Dans chaque vidéo, un opérateur interagit avec son environnement en ajoutant de nouveaux objets à la scène, en déplaçant quelques objets selon des instructions textuelles ou en retirant d'autres objets. Dans chaque enregistrement, 3 à 13 actions sont exercées. D'autre part, les vidéos sont enregistrées sous différents types de source d'illumination (séparément) : lumière du jour, éclairage fluorescent, LED et éclairage incandescent. De plus, la même caméra, installée sur un trépied, est utilisée pour filmer les vidéos avec des paramètres de captures adaptés à chaque scène.

Durée totale des vidéos	693 secondes	**Nombre d'actions**	109 actions
Durée totale en trames	20112 trames	**Durée totale des actions**	8932 trames
Durée minimale des vidéos	500 trames	**Durée minimale d'action**	17 trames
Durée maximale des vidéos	2950 trames	**Durée maximale d'action**	530 trames
		Durée moyenne d'action	81.9450 trames

Tableau 4 : détails du corpus.

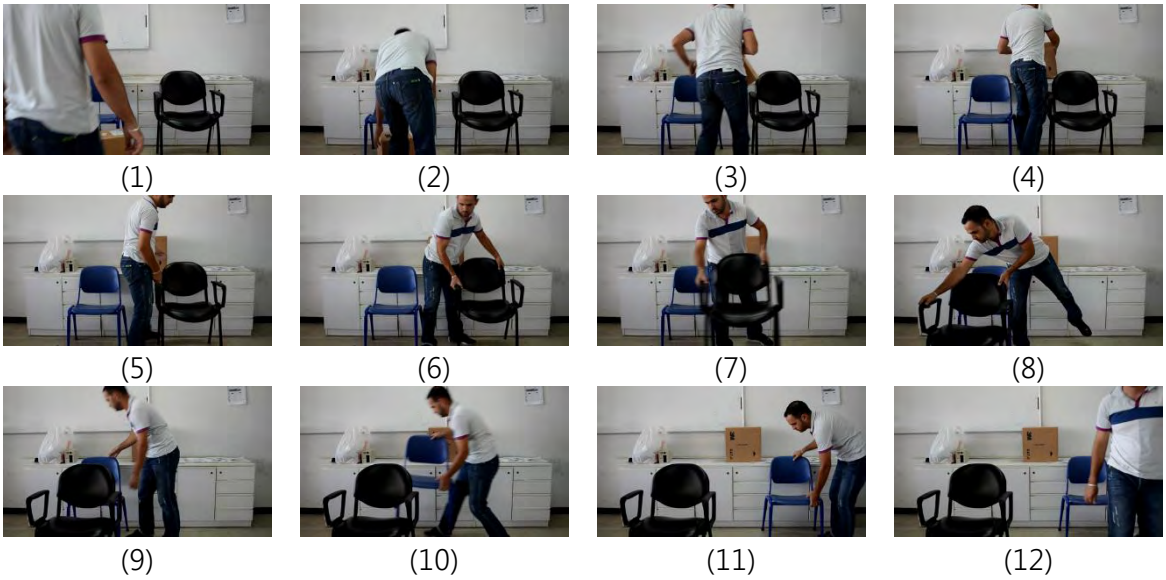


Figure 32: Contenu d'une vidéo à trois actions (prendre le carton et le poser sur la table blanche, déplacer le fauteuil noir, déplacer la chaise bleue).

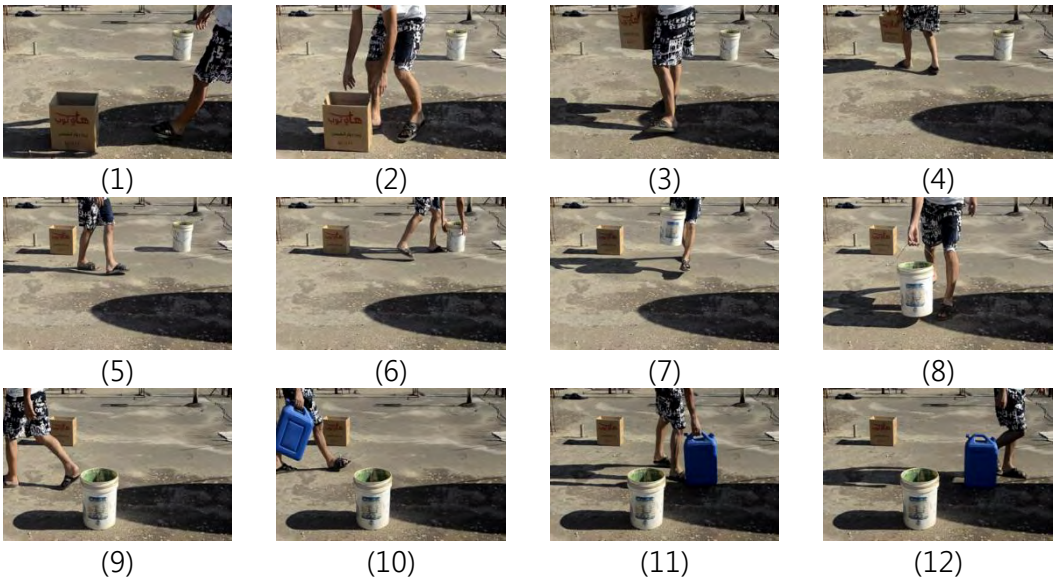


Figure 33: Contenu d'une autre vidéo (reculer le carton, approcher le bidon, amener le jerrican.

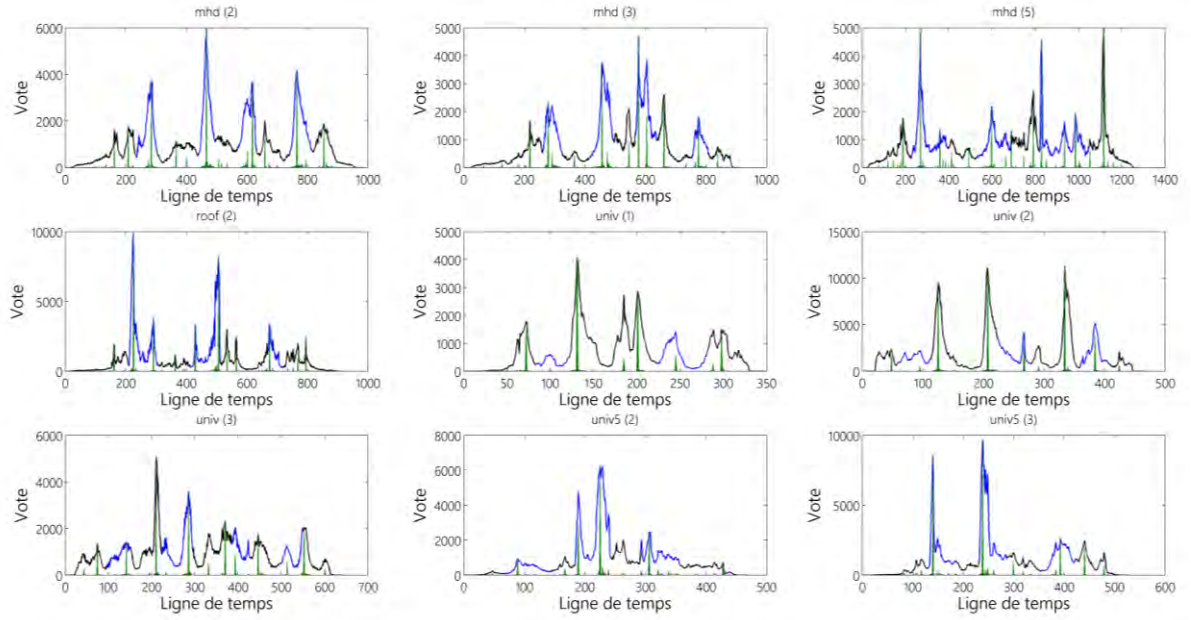


Figure 34: Vecteurs de votes (bleu et noir) et des maximums (vert) correspondant à neuf vidéos. Les segments bleus dans les vecteurs des votes indiquent les segments vidéo qui entourent une action (selon la vérité terrain). On peut facilement remarquer que les positions ayant un vote élevé correspondent aux positions des limites réelles d'actions (les extrémités des segments bleus).

3.6.2 Méthode d'évaluation

L'évaluation des résultats obtenus consiste à comparer les limites produites automatiquement avec les limites de la vérité terrain. Les limites de la vérité terrain sont des limites localisées manuellement par un opérateur expert dans le but de former une référence de comparaison. Étant donné la séquence des images formant la vidéo, les limites sont supposées représenter les instants auxquelles une action commence ou se termine du point de vue de l'opérateur.

En général, le choix de la méthode d'évaluation dépend des types des résultats de la méthode. Dans cet outil, nous ne détectons pas les segments qui entourent une action, mais nous détectons des points temporels comme étant des positions candidates des limites d'action. C'est donc l'exactitude du positionnement d'un point dans le temps et non d'un segment complet que nous cherchons à mesurer.

D'autre part, nous observons que pour ce type de sujet (les limites d'action), les annotations manuelles effectuées par plusieurs opérateurs humains ne sont pas toujours identiques, mais varient dans un petit intervalle. Cette variation est due à l'ambiguïté du moment exact où une action commence ou se termine. Cette imprécision sera prise en considération en évaluant les limites produites par un opérateur humain indépendant (autre que nous) de la même manière que seront évalués les résultats automatiques.

Nous proposons une méthode pour évaluer la capacité de notre outil de proposer des points temporels situés dans le voisinage des limites de la vérité terrain ("**VT**") qui repose sur le calcul des distances entre les limites automatiques et les limites de la référence (Limites "**VT**"). L'analyse de ces distances donne une indication sur la fiabilité et la précision des limites détectées. Le résultat attendu est l'observation des distances qui soient, en général, petites.

Nous proposons de calculer la distribution de deux groupes de distances :

→ **"Auto-to-REF" :** Distances entre chaque limite automatique et la limite "**VT**" la plus proche. Ces distances reflètent la précision des limites produites.

→ **"REF-to-Auto" :** Distances entre chaque limite "**VT**" et la limite automatique la plus proche. Ces distances montrent la capacité de localiser toutes les limites "**VT**".

Un système automatique qui retournerait des limites d'action regroupées autour d'une seule limite de la vérité terrain (Figure 35) va produire des valeurs faibles pour le premier type de distance, mais élevées pour le second. On peut également identifier un cas symétrique où les limites de la vérité terrain seraient concentrées et toutes proches d'une limite automatique (Figure 36) produisant une faible valeur des distances du deuxième type alors que les distances du premier type seraient bien plus élevées.

Dans le cas idéal, nous attendons des distributions équilibrées des distances de deux types, et autant que faire se peut, concentrées autour de 0.

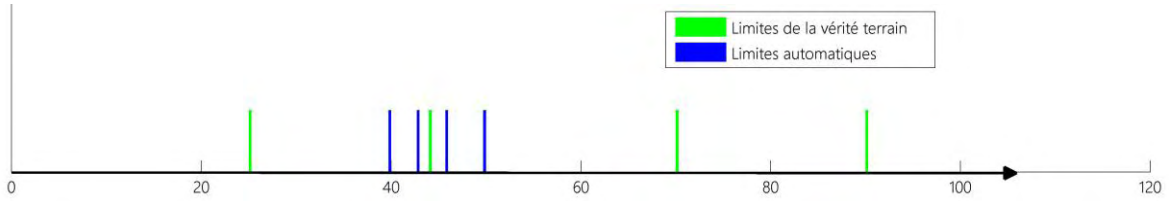


Figure 35 : cas où un système automatique retourne des limites d'action regroupées autour d'une seule limite de la vérité terrain.

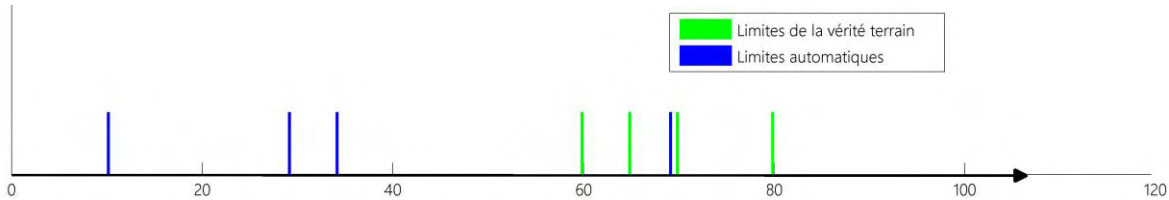


Figure 36 : cas où les limites de la vérité terrain seraient concentrées et toutes proches d'une limite automatique.

Pour produire une illustration graphique des résultats, nous calculons l'histogramme des distances quantifiées linéairement sur 10 niveaux Dans les figures suivantes, chaque intervalle de quantification est représenté sur l'axe d'abscisse par son milieu. L'axe des ordonnées montre le nombre des distances associées à chaque intervalle.

Ensuite, nous normalisons les nombres de distances en les divisant par le nombre total de ces distances calculées. L'objectif de ces figures étant de montrer que les deux groupes des distances sont formés principalement par de petites distances, nous souhaitons observer deux courbes en forme de "demi-Gaussiennes".

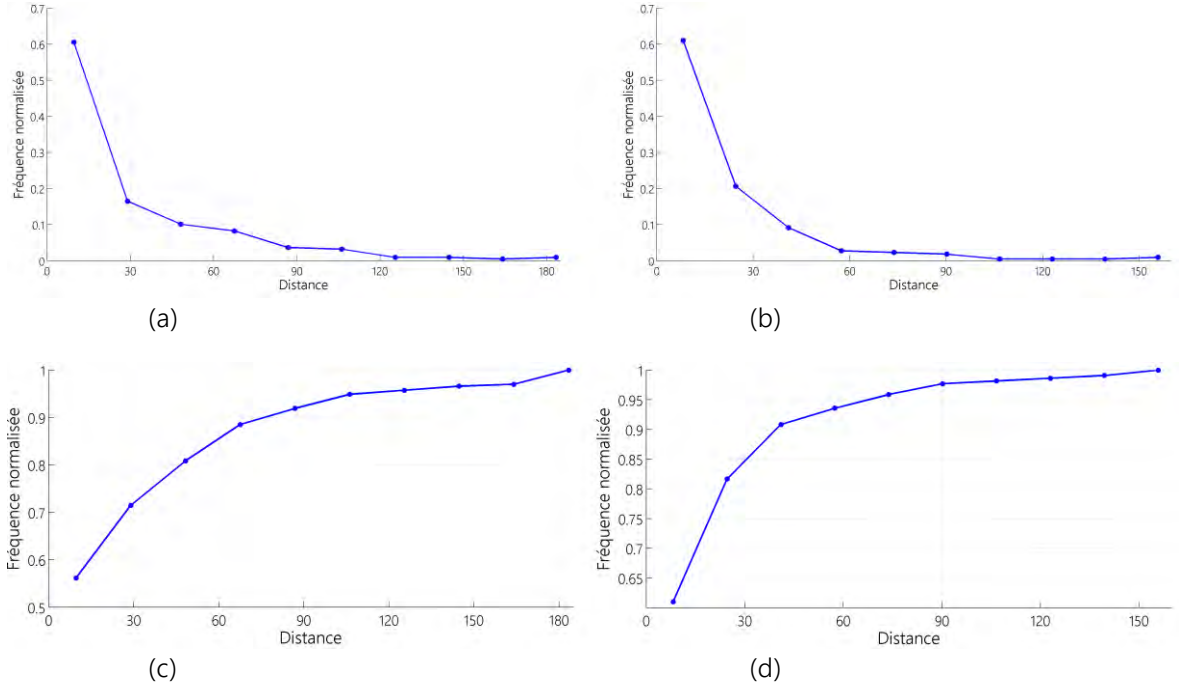


Figure 37: Histogrammes des distances en trames. (a) Groupe des distances "Auto-to-REF", (b) Groupe des distances "REF-to-Auto", (c) histogramme cumulé du groupe "Auto-to-REF", et (d) histogramme cumulé du groupe "REF-to-Auto"(Les détails du corpus utilisé sont décrits dans le Tableau 4).

Les actions détectées par notre système	
Nombre des limites	218 limites
Distances du groupe "Auto-to-REF"	
Moyenne	33.0383 trames
Écart-type	40.6662
Distances du groupe "REF-to-Auto"	
Moyenne	21.2844 trames
Écart-type	25.4233

Tableau 5 : Moyennes et écart-types des distances.

Nous remarquons que les graphes (a) et (b) de la Figure 37, sont bien de forme "demi-Gaussienne", car la plupart des limites proposées sont proches des limites "VT". Et, selon le graphe (c), 90% (axe vertical) des limites automatiques sont à une distance 80 trames d'une limite "VT". Et d'après la figure (d), 95% (axe vertical) des limites "VT" sont associées à au moins une limite automatique située à une distance inférieure à 80 trames. Toutefois, l'intérêt de ces résultats est limité par l'absence d'éléments de comparaison avec d'autres systèmes. Nous ne connaissons pas de travaux qui visent à

résoudre un problème similaire, et par conséquence, cette comparaison semble impossible. Pour contourner cette limitation, nous proposons de concevoir 3 systèmes de référence :

1) *Un système "Oracle"*
   Ce système est considéré comme étant un système idéal qui produirait les limites les plus fiables. Pour nos expérimentations, ces limites sont localisées manuellement par un opérateur humain (autre que celui qui a produit la "VT"). Cet opérateur reçoit comme instruction la demande de préciser les instants qu'il considère comme étant des débuts ou des fins d'actions. La différence entre l'oracle et la "VT" montre l'impossibilité de construire une réponse exacte concernant les positions des limites, et donc que celles-ci relèvent en partie d'une appréciation subjective.

2) *Programme de base 1*
   Ce système est supposé traiter le problème de manière simpliste et naïve, quitte à produire de mauvais résultats. Nous considèrerons ici un système qui choisit les limites de manière aléatoire. Ce système ne repose sur aucune information liée au contenu de la vidéo. Dans ce système, les deux contraintes imposées, de manière à autoriser les comparaisons, est que le nombre de limites produites doit être égal à celui de notre système et que la distance séparant deux limites consécutives soit comprise dans l'intervalle défini par la distance minimale et la distance maximale entre deux limites de la vérité terrain

3) *Programme de base 2*
   De la même manière que précédemment, nous souhaitons pouvoir comparer nos résultats avec ceux qui seraient produits par un programme « naïf » qui chercherait à résoudre le même problème. Nous utiliserons pour cela un système de détection de changements de plans. Même s'il n'est pas conçu et implémenté pour détecter les limites des actions, nous formulerons l'hypothèse que dans certains cas, la modification apportée à la scène par une action peut ressembler à un changement progressif ou radical du contenu de l'image, comme c'est le cas lors d'un effet de transition. Bien entendu, nous ne prévoyons pas de bons résultats sur la détection des limites d'actions. Pour avoir une comparaison raisonnable, nous avons néanmoins réglé les paramètres de ce système de manière à obtenir un nombre de limites proposées égal au nombre de limites détectées par notre système. Nous avons utilisé le système de détection des changements de plans implémenté dans la toolbox "Computer Vision System" de MATLAB (65).

Le graphe ci-dessous montre que nos résultats se situent entre ceux de ces systèmes.

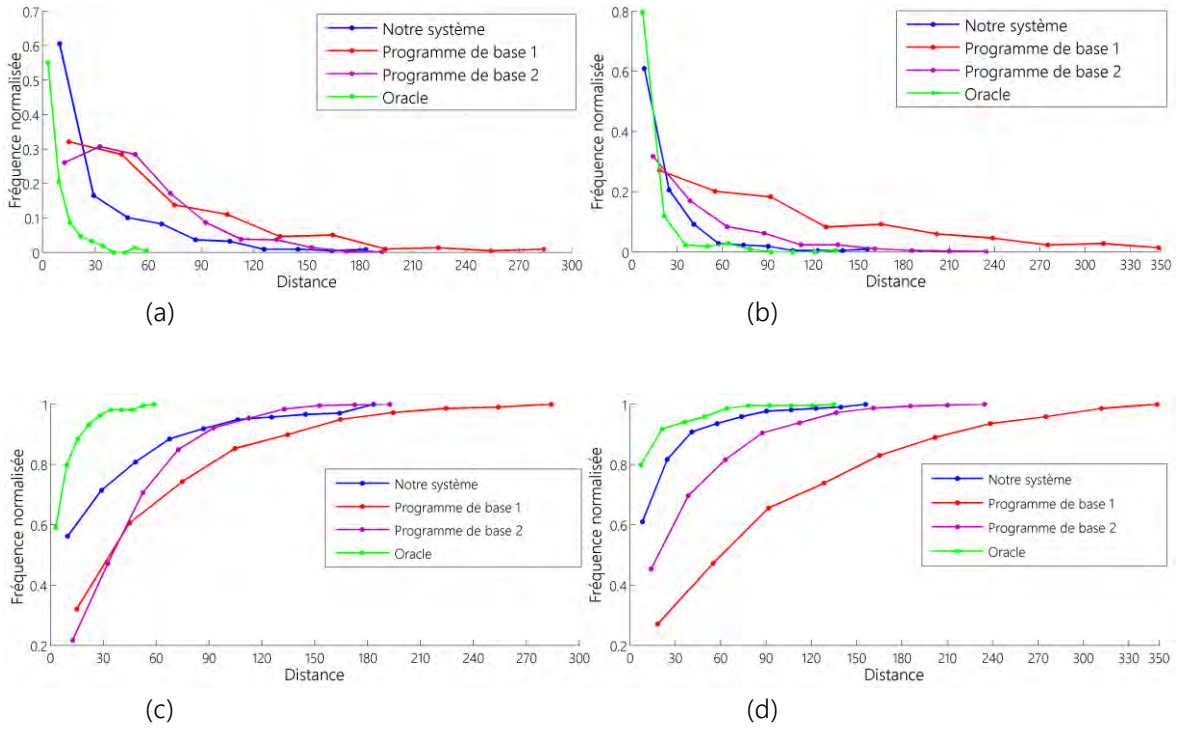


Figure 38 : Comparaison avec les systèmes : "Oracle", "programme de base" et "programme de base 2". (a) et (b) : Les fréquences normalisées des distances calculées pour chaque système ((a) : groupe "Auto-to-REF", (b) : groupe "REF-to-Auto"). (c) et (d) : Histogrammes cumulés normalisés des distances obtenues par chaque système ((c) : distances du "Auto-to-REF", (d) : distances du "REF-to-Auto").

Sur ces graphiques, un système apparaît d'autant plus performant que la courbe associée est proche de l'axe vertical. Nous observons que la courbe de notre système est plus proche de celle de l'oracle que de celle des programmes de base.

Nous remarquons que dans les deux graphiques (a) et (b) de la Figure 38, les distances de notre système sont distribuées sous une forme assimilable à une demi gaussienne qui est plus proche de l'axe des ordonnées que celles du "programme de base". Autrement dit, les grandes distances entre résultat automatique et vérité terrain (à droite de l'axe des abscisses) dans la courbe des systèmes "programme de base 1" et "programme de base 2" sont plus fréquentes qu'avec notre système. Inversement, cela signifie que les limites détectées par notre système sont plus proches des limites réelles qu'avec les systèmes "programme de base" et "programme de base 2".

	Distances du "Auto-to-REF" (en trame)		Distances du "REF-to-Auto" (en trame)	
	Moyenne	**Écart-type**	**Moyenne**	**Écart-type**
Oracle	8.3165	9.7544	11.0734	17.2037
Notre système	33.0383	40.6662	21.2844	25.4233
Programme de base 1	63.7018	57.7571	100.7844	84.6882
Programme de base 2[2]	45.0628	41.5802	36.6938	39.1214

Tableau 6 : Moyennes et écart-types des distances produites par chaque système, sur le corpus décrit dans le Tableau 4.

L'efficacité du système proposé peut également être mesurée en termes de "Rappel" et de "Précision". Il n'est pas très pertinent d'évaluer la capacité de notre système à détecter avec précisions les limites identifiées dans la vérité terrain (**VT**), car même des segmentations manuelles peuvent être différentes d'une personne à l'autre (cf. Oracle). Pour cela, on compare les capacités des systèmes à détecter une limite située dans le voisinage des limites réelles. Dans le tableau suivant (Tableau 7), on calcule la précision et le rappel des limites détectées par chaque système, en supposant chaque fois qu'une limite située à une distance inférieure à un seuil est considérée comme correcte.

Système	**Oracle**		**Notre système**		**Programme de base**		**Programme de base 2**	
Seuil (Nombre des trames)↓	**Précision**	**Rappel**	**Précision**	**Rappel**	**Précision**	**Rappel**	**Précision**	**Rappel**
< 15	0.9000	0.9000	0.4810	0.7798	0.3165	0.5688	0.3851	0.7119
< 30	1	1	0.6646	0.9450	0.5570	0.7615	0.6102	0.8713
< 45	1	1	0.7405	0.9908	0.6519	0.8761	0.7088	0.9610
< 60	1	1	0.8228	1	0.7405	0.9312	0.7913	0.9924
< 120	1	1	0.9684	1	0.9177	0.9954	0.9402	1
< 180	1	1	0.9876	1	0.9620	1	0.9827	1

Tableau 7 : mesures de précision et de rappel sur les limites détectées par chaque système.

Au cours des évaluations ci-dessus, nous n'avons considéré que les positions des limites automatiques proposées. Dans ce qui suit, nous étudions la dépendance entre

[2] Les résultats de ce système sont obtenus sur une partie du corpus pour réduire le coût de calcul.

la fiabilité d'une limite détectée et la valeur du vote associé. Cette étude vise à montrer qu'une limite est d'autant plus fiable qu'elle est associée à un vote élevé.

Durant cette évaluation, nous commençons par normaliser les votes des limites détectées dans chaque vidéo de sorte que le vote maximal soit égal à 1. Ensuite, nous divisons l'ensemble des valeurs des votes obtenues en 10 sous-ensembles de sorte que chaque sous-ensemble comporte le même nombre de vote. Le premier sous-ensemble correspond aux 10% des votes le plus faibles et le dernier correspond aux 10% des votes les plus forts. Pour chaque sous-ensemble, nous calculons la valeur du vote moyen, et la valeur de la distance moyenne à la limite GT la plus proche. Dans la Figure 39, l'axe d'abscisse représente les sous-ensembles de votes ordonnés de gauche à droite d'une manière croissante (les distances correspondantes aux votes les plus élevés sont à droite). L'axe d'ordonné représente la moyenne normalisée des distances de chaque sous-ensemble.

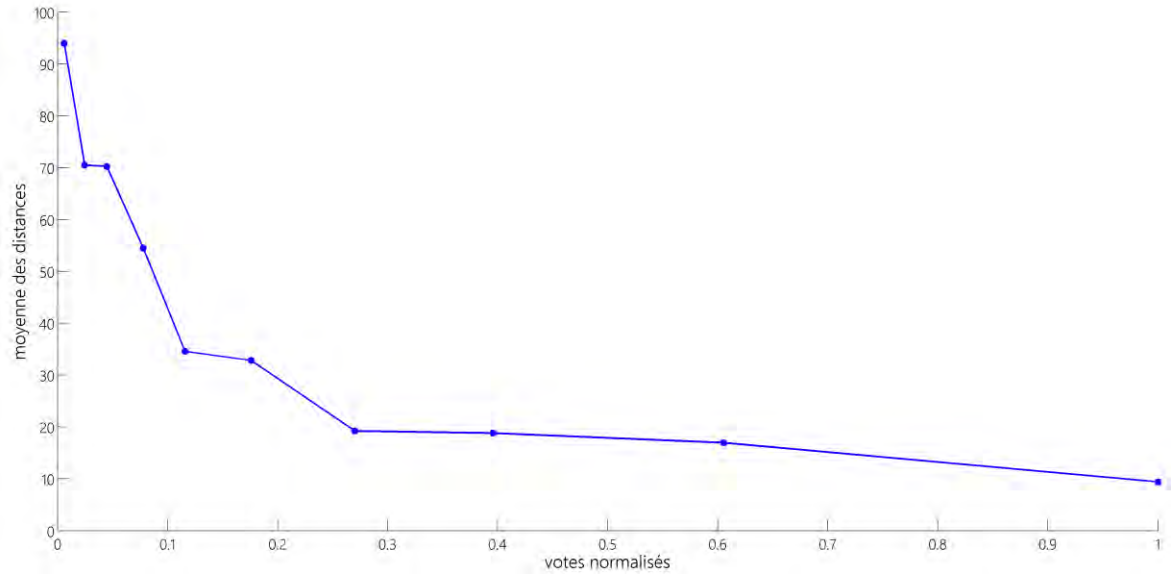


Figure 39: Moyenne des distances qui correspondent à chaque sous-ensemble de votes.

Nous observons clairement que les votes les plus élevés sont associés à des limites automatiques situées à de petites distances des limites réelles.

3.6.3 Facteurs agissant sur le résultat

Les études qui précèdent nous conduisent à déduire que le système proposé se comporte bien sur les cas et les environnements étudiés (sur une précision fixée à 3 secondes). Cependant, nous ne pouvons pas généraliser cette conclusion à n'importe quel autre cas. Plusieurs facteurs agissent sur la performance du système :

1) *l'arrêt de l'opérateur*
   Notre système est basé sur l'analyse des valeurs historiques des pixels afin de localiser des modifications persistantes d'une couleur. Cependant, le cas d'un

opérateur qui s'arrête et reste immobile pendant un long moment ne peut pas être différencié d'une action exercée sur l'arrière-plan.

2) *l'occultation partielle ou complète du déroulement d'action*
Les limites ne peuvent pas être détectées dans les cas où l'action exercée est cachée par l'opérateur ou un autre objet. Malgré cela, le système reste susceptible de détecter au moins une limite (la fin d'action) à l'instant où l'effet de l'action devient visible par la caméra.

3) *la dimension de l'objet soumis à l'action*
Dans le cas où l'objet soumis à l'action est proportionnellement très petit dans la scène, peu de pixels sont engagés dans la caractérisation de la modification, et par suite, l'impact dans le vecteur final des votes est négligeable.

3.7 Conclusion

Dans ce chapitre, nous avons proposé une méthode pour la localisation des limites d'actions effectuant une modification persistante sur l'environnement de l'opérateur. En se basant sur une méthode de quantification par Codebook initialement proposée pour soustraire l'arrière-plan, nous avons développé un automate de décision combiné à un système de votes pour extraire ces limites. Un vote élevé produit par cette méthode indique une limite candidate. Une méthode d'extraction des maximums de ce vecteur permet ensuite de trouver les positions des limites d'actions. L'absence de corpus offrant des contenus pertinents pour notre problème, nous a conduit à mener une étude sur un petit volume d'enregistrements adaptés. Ils ont ainsi servi de support pour comparer notre méthode avec un système de type « Oracle » et un programme de base. Les résultats généraux permettent de cerner les performances de cette méthode et ainsi d'en déterminer des modalités de mise en œuvre comme nous le verrons dans le Chapitre 5 : "Synchronisation".

Chapitre 4

DÉTECTION DES RÉPÉTITIONS

4.1 Introduction

Dans le chapitre précédent, nous avons proposé une méthode permettant d'identifier les limites potentielles d'actions individuelles dans une vidéo induisant des modifications persistantes sur l'environnement de l'opérateur. Cependant, le contenu textuel n'est pas formé exclusivement d'instructions visant à exécuter chacune une action unique. Certaines d'entre elles conduisent à répéter plusieurs fois une action de manière répétitive (répétitions en boucle) ou à des moments éloignés (répétitions séparées). Dans ce cas, le système de segmentation est susceptible de détecter les occurrences de l'action répétée comme étant des actions indépendantes. Or, l'information de l'existence d'une répétition dans le texte et dans la vidéo peut être d'un apport significatif au moment de mettre en correspondance les 2 contenus, comme nous le verrons plus loin. Sans remettre en cause pour autant l'intérêt des résultats de la segmentation en actions, nous allons essayer de trouver une solution – indépendante de cette segmentation – capable d'identifier :

→ Une répétition en boucle : nous cherchons alors un segment vidéo comprenant plusieurs "occurrences" identiques consécutives. Une "occurrence" dans ce cas peut être une action unitaire, un groupe d'actions, voire même une répétition en boucle (dans le cas de répétitions imbriquées).

→ Une répétition séparée : nous cherchons deux ou plusieurs segments disjoints dans le temps, dans lesquels la même action, ou la même séquence d'actions, ou la même boucle est exécutée.

Nous aborderons ces deux cas séparément, puis nous verrons comment combiner les deux méthodes proposées en un seul système.

Dans la section suivante, nous présentons notre méthode de détection des répétitions d'actions en boucle dans une vidéo. Nous montrons comment il nous est possible d'extraire des paramètres décrivant cette répétition comme le nombre et la durée des occurrences. Cette méthode repose sur une estimation de la fréquence fondamentale (souvent notée F0) initialement conçue pour l'analyse d'un signal audio, et que nous avons dû adapter à notre problème.

À la suite, nous présentons une adaptation d'une méthode de détection des segments répétés dans un flux télévisé, pour la détection d'actions répétées de manière séparée.

Finalement, nous argumentons l'intérêt des méthodes proposées et indiquons comment elles sont intégrées dans notre système global de synchronisation.

4.2 État de l'art

Nous avons trouvé très peu de travaux dans l'état de l'art sur l'analyse des contenus vidéo qui abordent le problème de la détection des répétitions. Alors que nous cherchons ici à repérer des segments qui contiennent une action répétée (c'est à dire des occurrences consécutives d'un même segment, comme dans le cas d'une répétition en boucle). Les quelques travaux que nous avons identifiés visent à détecter des segments répétés à différents moments dans la vidéo ou dans plusieurs enregistrements.

Nous présentons dans cette section des travaux similaires rassemblés en trois groupes : ceux qui traitent de la structuration de flux vidéo, ceux qui traitent de la détection des messages publicitaires, et ceux qui abordent le problème de la détection de copie.

4.2.1 Structuration de flux télévisé

La plupart des travaux qui portent sur la détection des segments répétés ont porté sur l'analyse du flux télévisé. L'analyse de ce type de flux développe toute une problématique autour de la diversité du contenu. Le filtrage des informations importantes, ainsi que la segmentation et l'identification des programmes diffusés, sont l'objet de la plupart de ces travaux.

Dans un flux télévisé, il est possible de distinguer les contenus répétitifs qui sont des segments vidéo diffusés plusieurs fois de manière identique, comme c'est le cas pour les bandes annonces, publicités, identification des sponsors, etc… Cette nature

répétitive est une caractéristique exploitée par de nombreux travaux pour identifier les programmes de ce type.

4.2.1.1 Détection des bandes annonces

Comme on l'a mentionné, ce qui identifie les bandes annonces dans le flux télévisé est leurs répétitions fréquentes, leurs durées et la nature de leur contenu. Citons parmi les travaux les plus référencés, ceux qui s'appuient principalement sur des caractéristiques comme l'existence de trames de silence, la durée de la bande annonce, et une information sur le mouvement (21), (20) et (12). D'autres travaux reposent essentiellement sur la fréquence des répétitions dans les bandes annonces (24), (9) et (8). Ces travaux utilisent une technique de table de hachage où une séquence répétée est identifiée par l'existence de codes déjà existants dans la table de hachage. Différentes techniques pour identifier ces répétitions de code existent (6).

Comme on a vu, les travaux basés sur la détection des bandes annonces cherchent des segments qui se répètent d'une manière visuellement identique à des moments distants dans le temps. Par contre, dans notre travail, nous cherchons à détecter des segments qui contiennent une action répétée d'une manière consécutive. De plus, dans notre cas les occurrences ne sont pas toujours strictement identiques, contrairement aux répétitions de films publicitaires.

D'autre part, nous nous sommes intéressés au travail (2) dans lequel les auteurs proposent une méthode pour trouver les segments similaires dans une vidéo, en utilisant une représentation matricielle appelée : matrices de comparaison. Nous présenterons brièvement cette méthode à la fin de cette section, car nous y ferons référence dans la proposition de notre système de synchronisation à la fin de la thèse.

4.2.2 Détection de copies ("video copy detection")

En général, le but de la détection des copies est de limiter la violation des copyrights sur les contenus vidéo. Les techniques dans ce domaine analysent et interprètent le contenu d'une base de données audiovisuelle pour savoir si un segment donné est répété d'une manière identique ou modifiée. Parmi les problèmes abordés par ce type de travaux, nous trouvons la capacité de distinguer les copies d'un segment et les segments similaires, ainsi que celle de détection des copies qui ont subi des modifications (coupures, bruit additionnel, recadrage, *etc*...).

La plupart des travaux dans ce domaine utilisent un ensemble des descripteurs locaux ou globaux pour calculer une signature correspondant au segment cherché (la requête). Par suite, ils calculent les signatures des vidéos de la base de données. Les segments ayant la même signature que la requête sont considérés comme étant des copies.

Plusieurs techniques utilisent des descripteurs et des informations globales comme l'histogramme de couleurs, l'intensité, les limites de plan, le mouvement, etc... ((5), (13), (11), (22) et (23)). Les auteurs de (13) proposent une méthode pour trouver

les copies piratées sur Internet en calculant une signature s'appuyant sur les limites de plans. D'autres techniques utilisent des descripteurs calculés sur des régions locales et sur des points d'intérêt spatio-temporels (10) et (15). La technique généralement utilisée consiste à calculer la signature du segment requête en fonction de la caractéristique étudiée (distribution des couleurs, mouvement et texture), et ensuite à trouver les segments candidats dans la base de données. Une comparaison et une évaluation détaillée de ces méthodes est effectuée par (19).

Les travaux dans ce domaine s'appuient sur un segment requête pour en chercher des copies dans la base de données. Par contre, dans notre travail, nous ne disposons pas de segment requête car nous ne savons pas d'avance quel contenu est répété dans la vidéo. D'autre part, ces techniques sont conçues pour détecter des copies situées dans de grands contenus (flux TV) ou dans un grand nombre d'enregistrements différents (dans une base de données). Nous ne nous intéressons qu'à des répétitions dont les occurrences sont consécutives (au sein d'une même vidéo).

4.2.3 Détection de périodicité dans une vidéo

Une répétition est parfois vue comme étant un évènement périodique. Les méthodes conçues pour identifier la périodicité dans une vidéo, sont des méthodes candidates pour localiser une répétition.

(26) Proposent une méthode basée sur la Transformée de Fourrier et la fonction d'autocorrélation circulaire (ACF) pour trouver la période de la périodicité la plus forte. Ils extraient les périodes candidates à partir de la Transformée de Fourrier et puis les filtrent selon leurs valeurs et leurs positions dans le signal ACF. Mais le résultat de ce travail ne porte que sur l'estimation de la période sans chercher à localiser les répétitions.

(7) Proposent un système d'identification des mouvements périodiques comme le mouvement d'un pendule. Le système est décomposé en deux étapes : suivre et détecter l'objet en mouvement, puis calculer l'autosimilarité de cet objet lors de son évolution dans le temps. Une forte similarité entre les objets apparaissant à plusieurs intervalles de temps égaux conduit à déduire qu'il existe un mouvement périodique de période égale à la distance temporelle séparant ces instants. L'autosimilarité entre l'image de l'objet à chaque instant est représentée dans une matrice de similarité. Dans cette matrice, la périodicité apparaît sous forme de lignes parallèles à la diagonale. Un travail similaire est aussi proposé par (25) dans lequel les vecteurs de mouvement sont utilisés durant l'estimation de la périodicité.

Les auteurs de (18) ont vu la périodicité comme étant une collection des configurations 3D qui se répètent à une fréquence donnée. Après l'alignement de vidéos provenant de deux caméras indépendantes, des points d'intérêt spatio-temporel sont extraits de chaque trame. Les points choisis à chaque trame sont les points les plus significatifs pour représenter les variations de forme et le mouvement local (16).

Finalement, une analyse des positions et des mouvements de ces points conduit à estimer la période.

Les auteurs de (3) proposent une méthode pour la reconnaissance des gestes en détectant les mouvements périodiques dans la vidéo. Ils calculent la transformation de fourrier sur une séquence de valeur d'intensités des images, et puis ils utilisent l'amplitude du spectre pour détecter les régions qui contiennent un mouvement périodique, et la phase pour classifier les gestes dans les régions détectés. Dans une étude, (4) propose le calcul d'une matrice de similarité qui aide à détecter les gestes répétés dans une vidéo. La matrice est construite en calculant la différence entre la position du squelette – capturé par un capteur Kinect de Microsoft –à un moment donné et celle à un autre instant dans la vidéo. Les gestes similaires apparaissent sous forme de zones à faibles coefficients.

Les travaux ci-dessus peuvent être adaptés d'une manière ou d'une autre pour détecter les segments qui contiennent des actions répétées. En l'absence d'implémentations disponibles de ces méthodes, nous n'avons pas pu tester leurs capacités à détecter le type de répétitions qui nous intéressent.

4.2.4 Détection des actions répétées séparées

4.2.4.1 Introduction

Comme on l'a déjà mentionné au début de ce chapitre, le but est de proposer et mettre en place un système pour détecter les segments qui contiennent des actions répétées de manière consécutive (en boucle). Plus tard, dans le système de synchronisation des contenus textuels et vidéo, ces segments seront associés aux instructions définies comme étant des boucles.

Nous remarquons que la détection d'un autre type de répétition apparaît utile durant le processus de synchronisation : la détection des actions répétées à des moments différents dans la vidéo. Être capable d'identifier les segments qui contiennent l'exécution de la même action sera utile plus tard durant le processus de synchronisation, spécialement quand le contenu textuel contient des instructions différentes mais qui exigent l'exécution de la même action. Dans ce cas, une correspondance peut être effectuée entre l'ensemble des segments (vidéo) ayant des contenus similaires et l'ensemble des d'instructions similaires (texte).

4.2.4.2 Aperçu de la méthode de (2)

Les auteurs de (2) proposent une méthode pour identifier les segments similaires dans deux vidéos. Cette méthode est appelée : Intersection Quadratique Récursive - IQR). Dans cette méthode, les auteurs comparent deux contenus vidéo en se basant sur des caractéristiques visuelles de bas niveau. La vidéo est représentée par plusieurs séquences des valeurs. La comparaison des séquences de deux vidéos produit une matrice appelée : "**Matrice de comparaison**". Dans cette matrice, la valeur à l'entrée (i, j) représente le "**taux de couverture**" entre le segment i dans la séquence associée à la

première vidéo et le segment j dans la séquence associée à la deuxième vidéo. Une valeur élevée de cette entrée indique que les segments i et j sont fortement similaires (du point de vue de la caractéristique utilisée).

Les matrices de comparaison calculées pour chaque caractéristique sont fusionnées dans une seule matrice finale. Dans cette matrice, les segments similaires apparaissent sous forme de lignes de valeurs fortes qui sont parallèles à la première diagonale de la matrice (Figure 40). La détection de ces lignes conduit à identifier les segments similaires dans les deux vidéos. Dans notre travail, nous avons besoin d'une méthode simple pour détecter les segments similaires dans une même vidéo. Pour cela, nous proposons d'utiliser la même méthode pour calculer la matrice de comparaison mais en comparant la vidéo avec elle-même.

Le choix de cette méthode précisément vient du fait qu'elle permet de repérer les segments similaires d'une manière simple. D'autres méthodes de détection des segments similaires pourraient également être utilisées.

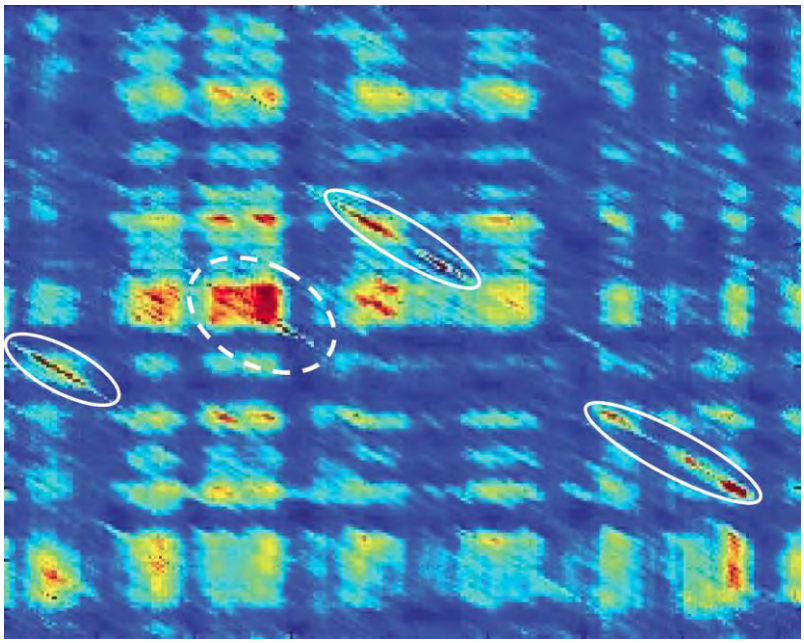


Figure 40 : Matrice de comparaison de deux plages publicitaires. Les ellipses indiquent les segments identiques ou similaires (2).

L'implémentation que nous avons faite de la méthode pour la détection des segments identiques, ne nous a pas toujours permis de détecter les segments qui contiennent les exécutions de la même action (les exécution de la même action ne sont pas toujours effectuées de la même façon). Les performances se sont montrées insuffisantes pour une exploitation immédiate – même à titre expérimentale – dans le cadre de notre travail. Afin de développer les travaux sur la synchronisation, nous formulerons l'hypothèse suivante :

Hypothèse V. *Nous supposons que nous disposons d'un outil capable de localiser les segments qui contiennent les exécutions de la même action.*

Dans ce qui suit, nous supposons que cet outil est capable de fournir des ensembles de segments qui contiennent les exécutions de la même action. Plus tard dans le système de synchronisation, chaque ensemble d'instructions similaires sera associé à un ensemble de segments similaires.

4.3 Méthode du YIN (Cheveigné et *al.* (1))

Nous considérons par la suite qu'une répétition en boucle est la répétition d'une ou plusieurs actions de manière consécutive dans une vidéo. Pour une répétition en boucle, nous cherchons à identifier le segment vidéo qui ne contient que les occurrences de cette répétition. L'occurrence peut être composé d'une seule action unitaire, un ensemble d'actions ou une autre répétition en boucle (appelée sous-répétition).

La détection de segments répétés dans une vidéo a fait l'objet de plusieurs travaux, comme nous l'avons évoqué dans le deuxième chapitre. Dans la plupart de ces travaux, les auteurs cherchent les apparitions d'un segment requête dans une vidéo, comme des spots publicitaires, des copies d'enregistrement dégradées ou rééditées, etc... En général, ces travaux se rapprochent d'avantage du second cas de répétition que nous avons identifié (c'est-à-dire les répétitions séparées). Mais ce qui les diffère de notre travail ici, c'est que nous cherchons la répétition d'une action "inconnue" et non-pas celle d'un segment requête. À notre connaissance, il n'existe pas de travaux publiés sur la détection d'actions en boucle, ce qui a motivé la recherche d'une solution originale à ce problème.

À l'inverse, la détection de segments répétés dans un signal audio a fait l'objet de l'attention de plusieurs chercheurs (42), (43), (44) et autres. Parmi ces travaux, une méthode – dite du YIN - qui estime la fréquence fondamentale F0 dans un signal audio, a attiré notre attention. Cette méthode vise à trouver la durée d'une période dans un extrait d'un signal audio plus ou moins périodique. Une certaine robustesse au bruit, et la grande lisibilité des résultats nous ont conduits à choisir cette méthode pour construire notre proposition.

4.3.1 Problématique et but de la Méthode YIN

La méthode appelée YIN, est une méthode proposée par Cheveigné et *al.* (1) comme étant une partie d'une solution globale visant à estimer la fréquence fondamentale d'enregistrements de musique et de parole avec une très faible erreur. Cette solution est basée sur un calcul d'autocorrélation entre le signal et lui-même

après un décalage de phase croissant. Selon les auteurs, cette proposition réduit considérablement l'erreur de l'estimation de la fréquence fondamentale.

Nous allons donc adapter cette méthode aux données de notre problème. Dans cette section, nous présentons tout d'abord la méthode de YIN, ensuite nous présentons quelques courbes résultant de son application afin de bien comprendre son exploitation dans notre système.

4.3.2 La méthode

Étant donné le vecteur $\mathbf{X=\{x_1, x_2, \dots x_N\}}$ représentant les valeurs d'un signal donné, échantillonné à intervalle de temps régulier, où $\mathbf{x_t}$ est la valeur du signal à l'instant **t** et **N** est le nombre de valeurs du signal. Dans le cas où le signal est parfaitement périodique de période **T**, l'équation suivante est toujours vraie à chaque instant **t**.

$$x_t - x_{t+T} = 0; \quad \forall t$$ **Équation 7**

Par suite, l'équation suivante est aussi vraie en considérant une fenêtre de taille **W**:

$$\sum_{j=t}^{t+W} \left(x_j - x_{j+T}\right)^2 = 0$$ **Équation 8**

En conséquence, une période inconnue peut être trouvée à l'aide de la fonction suivante :

$$d_t(\tau) = \sum_{j=t}^{t+W} \left(x_j - x_{j+\tau}\right)^2$$ **Équation 9**

La période cherchée est alors la valeur de $\boldsymbol{\tau}$ qui minimise la fonction distance. En particulier, si le signal est parfaitement périodique, on doit avoir $\mathbf{d_t(\tau)=0}$. Cependant l'Équation 9 ne peut pas être utilisée telle quelle parce que d'une part dans un signal périodique, il peut exister une infinité de valeurs $\boldsymbol{\tau}$ qui minimisent la fonction, qui sont toutes les multiples de la période. D'autre part, la fonction $\mathbf{d_t(0)}$ est toujours égale à zéro. Pour pallier ce problème, les auteurs de la méthode YIN ont remplacé la fonction des différences (Équation 9) par la fonction : "moyenne cumulative normalisée de la fonction distance" :

$$d'_t(\tau) = \begin{cases} 1, & si\ \tau = 0, \\ d_t(\tau) \Big/ \left[1/\tau \sum_{j=1}^{\tau} d_t(j)\right] & sinon \end{cases}$$ **Équation 10**

Cette fonction diffère de la précédente par sa valeur initiale à "1" à la place de "0", et par le fait qu'elle donne une valeur plus petite que "1" si la valeur de $\mathbf{d_t(\tau)}$ est plus petite que la moyenne.

4.3.3 Analyse et compréhension des résultats produits

Le calcul de la fonction donnée à l'Équation 10 permet de trouver la période d'un signal périodique, comme celui d'une fonction "sinus" par exemple.

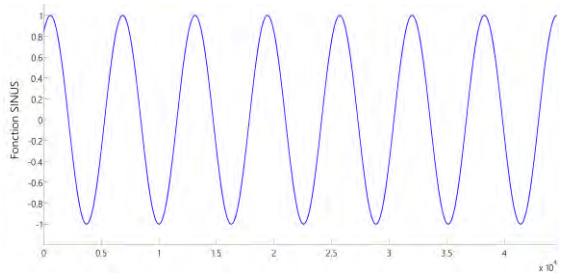


Figure 41: le signal de la fonction Sinus.

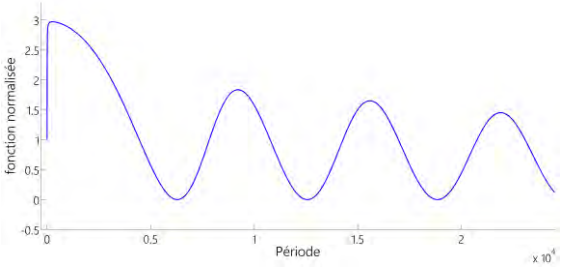


Figure 42: la fonction normalisée des différences.

Comme prévu, la Figure 42 montre que la fonction de moyennes cumulative normalisées des différences prend des valeurs à zéro pour des abscisses égales à la période de la fonction sinus ainsi qu'à ses multiples. Le premier zéro indique la période fondamentale du signal périodique.

Dans ce qui suit, nous présentons des cas où la méthode de YIN est moins pertinente et fiable.

4.3.3.1 Cas d'un signal partiellement périodique

Dans le cas où un segment périodique est entouré par d'autres contenus non périodiques, la fonction des différences échoue dans la caractérisation de la période exacte.

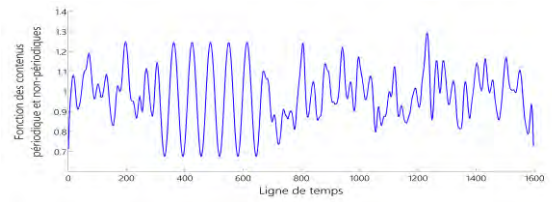


Figure 43: Exemple d'un signal formé d'une partie périodique entourée par d'autres contenus.

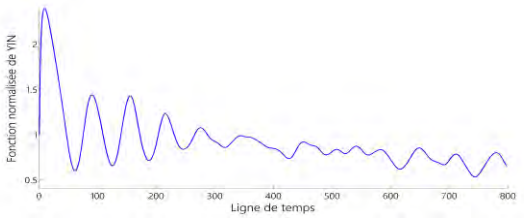


Figure 44: Fonction YIN normalisée correspondante à la Figure 43.

La Figure 43 présente un exemple d'un signal formé d'une partie périodique entourée par d'autres contenus. La fonction YIN normalisée correspondante est présentée dans la Figure 44. On peut facilement remarquer l'effet des contenus non-

périodiques sur le calcul de la fonction YIN en le comparant par celle de la Figure 42. Ceci affecte la position du minimum qui indique la durée de la période estimée. Dans cet exemple, le minimum est à la position 75 alors qu'il devrait être à la position 62.

4.3.3.2 Cas d'un signal contenant plusieurs segments périodiques

L'existence de plusieurs segments périodiques rend l'extraction des périodes plus complexe. Dans ce cas, chaque période produit potentiellement un minimum local de la courbe YIN différent. Il peut être alors difficile d'analyser la superposition des effets de ces segments sur le résultat global.

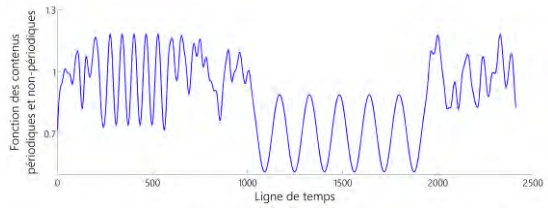


Figure 45: Exemple d'un signal formé de 2 parties périodiques entourées par d'autres contenus.

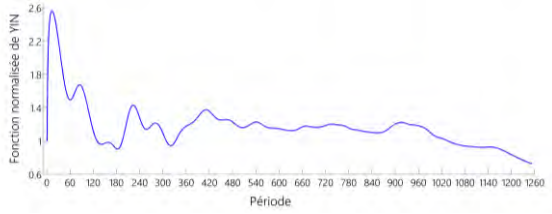


Figure 46: Fonction YIN normalisée correspondant à la Figure 45.

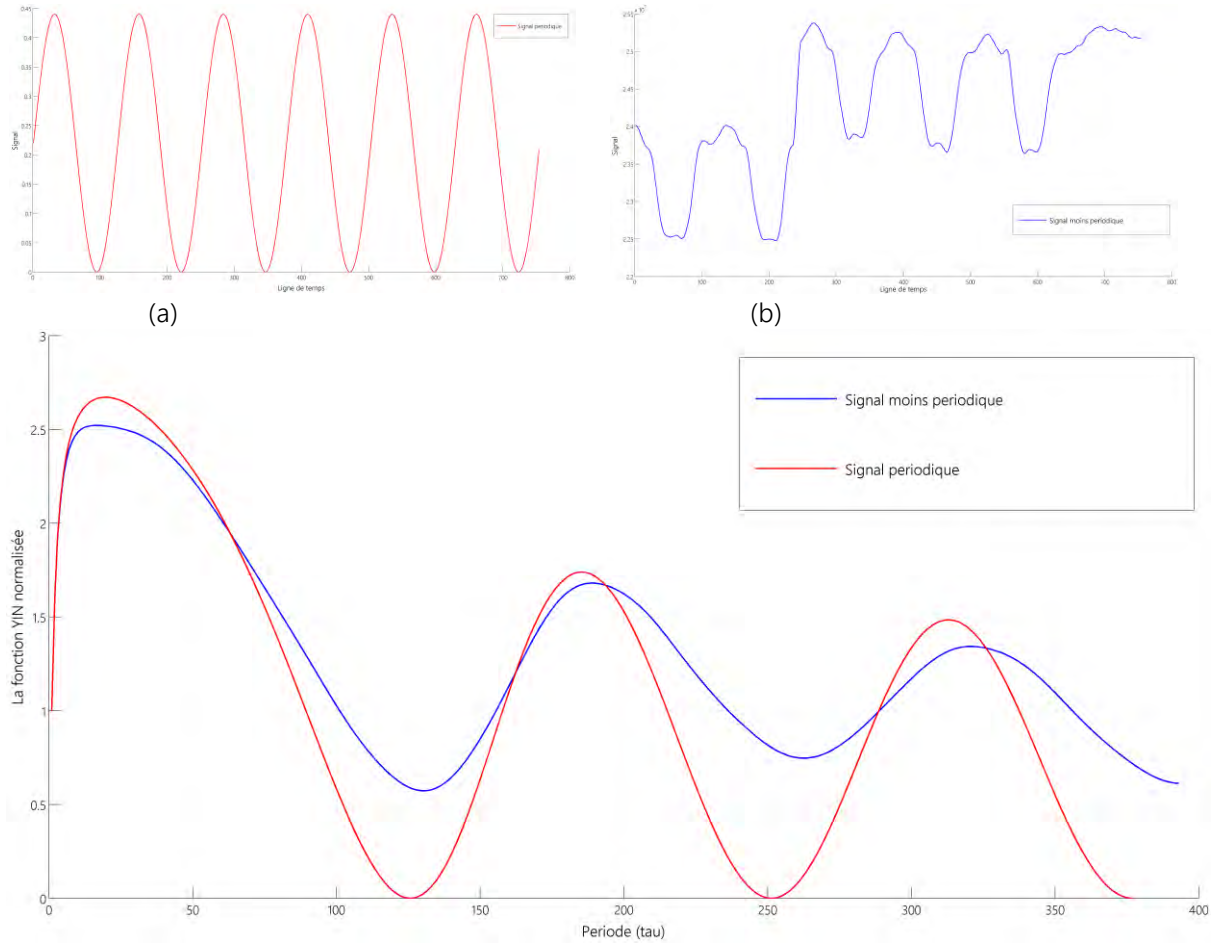
L'exemple de la Figure 45 présente un signal contenant deux segments périodiques différents. La Figure 46 montre la courbe de la fonction YIN normalisée correspondante. Nous devrions observer deux minimums locaux importants aux positions correspondantes aux valeurs des deux périodes (63 et 157), mais ce n'est pas le cas. Nous trouvons un minimum local proche de la première période, mais de valeur peu significative, et un autre au voisinage de la deuxième période, décalé de 20 images. De plus, plusieurs minimums locaux existent dans la courbe, ce qui rend l'identification automatique des périodes plus complexe.

4.3.4 Limitations

Notre objectif est de localiser toutes les répétitions dans une vidéo, si elles existent. Les auteurs de la méthode YIN avaient pour objectif de déterminer la période la plus vraisemblable d'un signal périodique en repérant la position du minimum le plus significatif sur l'abscisse du minimum. Cette information seule n'est pas suffisante pour plusieurs raisons :

→ Un signal non-périodique (et par extension une vidéo sans répétition) produira une courbe YIN qui comportera un minimum. Le problème posé est de pouvoir distinguer cette situation de celle ou le minimum renseigne sur la présence d'une répétition effective.

→ La méthode du Yin telle qu'elle ne renseigne pas sur la position où le signal est parfaitement périodique (lorsque celui-ci ne l'est pas complètement.



(c)

Figure 47 : relation entre la periodicité du signal et la valeur du minimum dans la courbe YIN. (a) est un exemple d'un signal parfaitement periodique, (b) est un exemple d'un signal moins periodique, et (c) montre les courve YIN correspondantes à (a) et (b).

On observe dans la Figure 47 que plus un signal se rapproche d'un signal périodique, plus le minimum de la courbe se rapproche de "0". Inversement, plus le signal s'éloigne d'un comportement périodique, plus ce minimum augmente. Nous exploitons cette propriété pour résoudre les problèmes évoqués plus haut.

4.4 Présentation générale de notre contribution

Notre plan de travail consiste au début à effectuer une mesure sur le contenu vidéo sensible aux répétitions pour générer un signal sur lequel on applique la méthode YIN. À partir du résultat, nous calculons une matrice dédiée qui permet de caractériser les répétitions, s'il y en a. Nous proposons une méthode d'analyse de cette matrice qui a pour objectif de localiser temporellement et d'extraire des paramètres informatifs sur ces répétitions.

4.5 Caractérisation des répétitions

Nous cherchons à produire une mesure sur les images successives d'une vidéo qui doit avoir un comportement caractéristique en présence de répétitions. Sélectionner une telle mesure doit tenir compte de la nature du corpus, des actions étudiées, *etc*...

Nous considérons les contenus vidéo où un opérateur interagit avec son environnement. Parfois, l'action consiste en une modification périodique de la scène (par exemple : accrocher plusieurs tableaux sur un mur) ; parfois, elle consiste en une répétition périodique d'un ensemble de mouvements spécifiques sans modification de l'environnement (par exemple : exécuter plusieurs séries d'un exercice sportif). Nous considérons de plus que la principale source de mouvement dans ces vidéos est l'opérateur.

Nous avons fait le choix de représenter chaque trame par son intensité dans la mesure où cette information permet de rendre compte des répétitions sur l'ensemble des vidéos que nous avons produites pour le corpus. Toutefois, la méthode que nous proposons est indépendante du choix de la caractéristique. De ce fait, toute autre caractéristique sensible aux répétitions peut être utilisée. Par exemple, on peut imaginer utiliser l'amplitude moyenne des vecteurs de mouvement pour des corpus dans lesquels le mouvement est particulièrement prégnant (Figure 48- a). Nous ne citons pas cette méthode pour des raisons de comparaison, mais pour indiquer que d'autres techniques sensibles aux répétitions peuvent être utilisées dans certains cas. Dans la suite de ce travail, une trame sera représentée par la somme des valeurs RGBs de ses pixels.

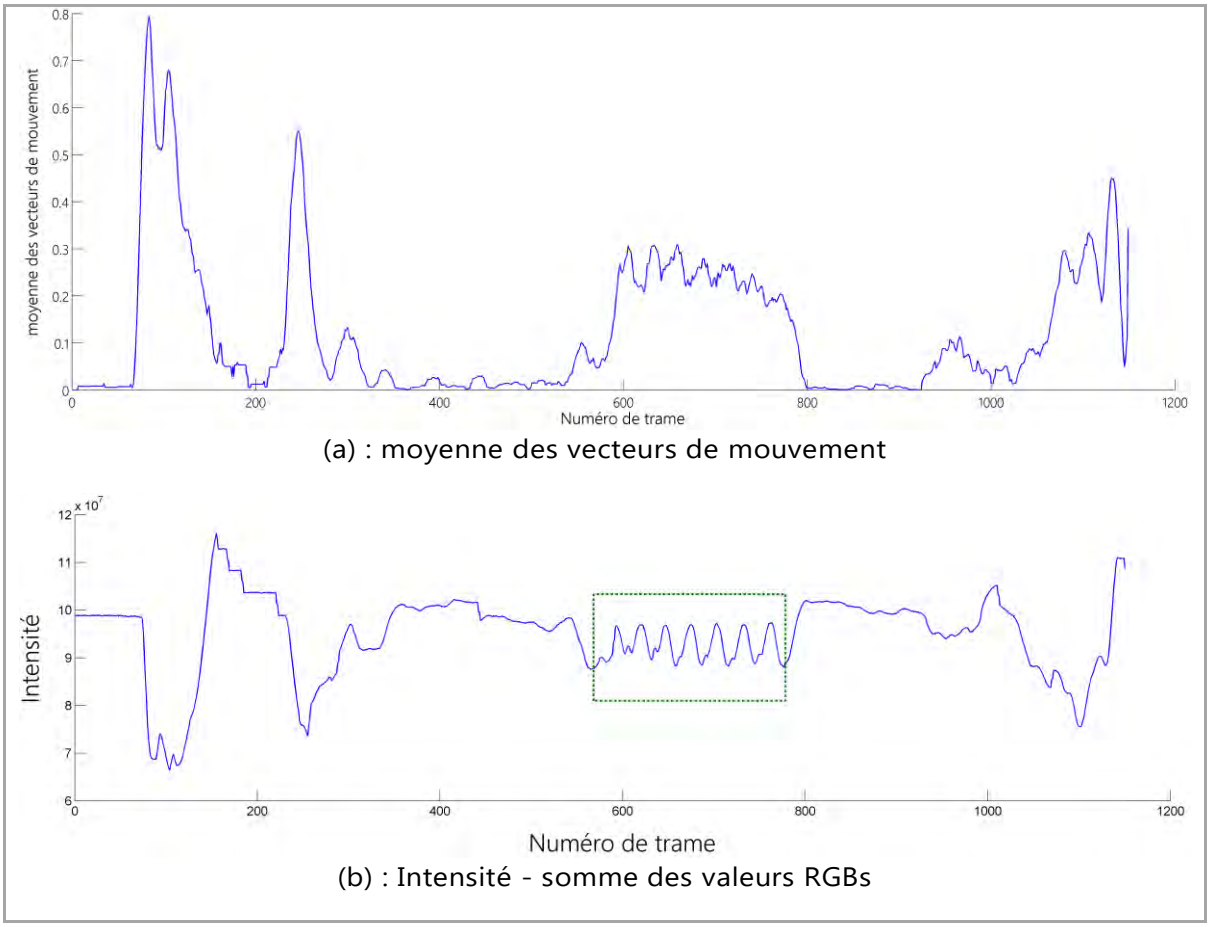


Figure 48: les représentations d'une vidéo réelle (les échantillons et les numéros de trame ci-dessus) contenant une action répétée 7 fois. On peut facilement localiser le segment périodique qui contient la répétition (rectangle vert) dans le signal d'intensité (b).

Le repérage visuel de la répétition est facile sur l'illustration de la Figure 48. Mais cette tâche est plus complexe et difficile à réaliser de manière automatique.

4.6 Matrice YIN

Nous proposons d'utiliser la fonction YIN comme étant l'inverse d'une mesure de similarité entre les occurrences d'une répétition. Cette fonction produit un minimum d'autant plus proche de "0" que la périodicité du signal analysé est parfaite. Par conséquence, le minimum produit par un signal qui ne contient aucune périodicité est plus grand que celui produit par un signal ayant une périodicité. En conclusion, utiliser le minimum de la fonction YIN comme étant la valeur d'une mesure (que nous appellerons "**my**") qui indique la possibilité d'avoir une répétition dans le signal forme la clé de notre solution.

Mais comme on a déjà indiqué, le calcul de la fonction YIN sur le signal entier ne permet pas de localiser une répétition. Pour cela, nous allons calculer le "**my**" sur des segments extraits du signal entier et susceptibles de contenir une répétition. La question qui se pose est : quels sont les segments susceptibles de contenir une

répétition ? À défaut d'une connaissance externe qui permettrait de cibler une plage temporaire a priori, la réponse dans notre cas est simple : tous les segments possibles.

Dans le contexte de notre problème, la répétition peut avoir lieu à n'importe quel moment dans la vidéo et elle peut durer sur un nombre indéterminé de trames. Nous allons donc calculer de manière exhaustive les "**my**" de tous les segments du signal (extraits à chaque position et de chaque durée possibles).

On définit la matrice **MATYIN** de la manière suivante:

$$\mathbf{MATYIN}_{i,j} = \mathbf{my}_{i,j}$$

Où, "$\mathbf{my}_{i,j}$" est le minimum de la fonction YIN calculé sur un segment de durée **i** qui commence à la trame **j**.

Équation 11

Comme nous pouvons le constater sur les exemples précédents, cette matrice est structurée et contient des zones de "relativement" petites valeurs qui correspondent aux répétitions. La forme géométrique et le repérage automatique de ces zones seront discutés dans la section suivante. Pour l'instant, on présente dans les figures suivantes les signaux de deux vidéos et leurs matrices **MATYIN** correspondantes.

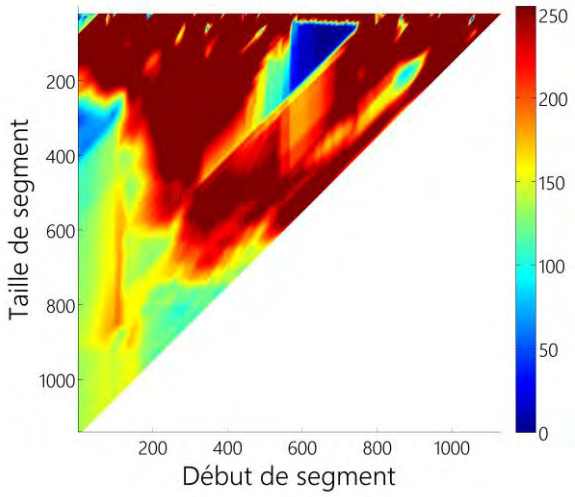


Figure 49: La matrice MATYIN correspondant au signal (Figure 48) d'une vidéo (Sport 6) contenant une répétition.

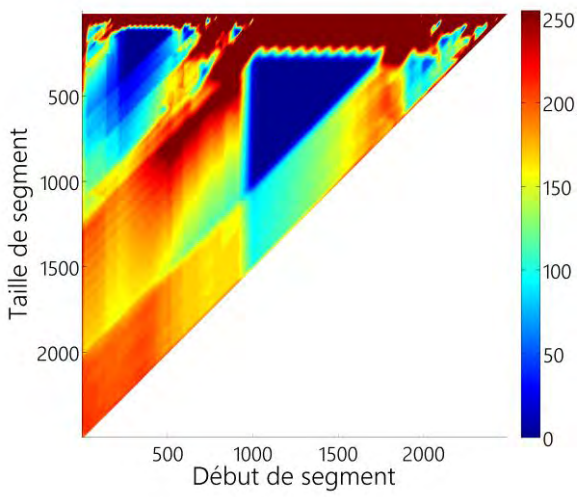


Figure 50 : La matrice MATYIN correspondant au signal (Figure 45) d'une animation contenant deux répétitions différentes et séparées.

La Figure 49 présente la matrice **MATYIN** d'une vidéo réelle qui contient une répétition isolée. Cette répétition correspond à une personne qui effectue une activité sportive répétitive (Série de sauts). Nous remarquons le triangle bleu qui est une zone de petites valeurs dans cette matrice. Cette zone correspond à la répétition.

La Figure 50 présente la matrice **MATYIN** d'un signal tiré d'une animation qui contient deux répétitions différentes et séparées dans le temps. Comme précédemment, les deux zones de faibles valeurs correspondent bien aux deux triangles présents dans **MATYIN**.

4.7 Localisation des répétitions

La localisation des répétitions dans une vidéo consiste alors à détecter les zones des petites valeurs dans la **MATYIN**, ainsi que des informations liées à leurs dimensions. Nous commençons par caractériser la forme géométrique de ces zones. Cette caractérisation nous permettra de définir une méthode dédiée qui vise à détecter ce type de forme. Nous déduirons ensuite des résultats diverses informations sur la répétition concernée.

4.7.1 Forme géométrique

Nous exprimons un jeu de paramètres décrivant une répétition donnée dans une vidéo. Soient :

- → **D** : la position du début,
- → **F** : la position de la fin,
- → **L** : la durée complète de l'enregistrement
- → **N** : le nombre d'occurrences de l'action répétée,
- → **O** : la durée d'une occurrence.

Les segments qui produisent une petite valeur **"my"** (périodique) sont situés entre le début et la fin de la répétition. Les tailles et les positions de ces segments sont définies par :

$$\mathbf{Seg = \{Signal\ (d \rightarrow d+l);\ Tel\ que\ [2xO <= l <= L]\ et\ [D<=d<=F-l]\}}^{3}$$

Équation 12

Selon le principe de **MATYIN**, **"l"** – qui est une taille de segment – correspond à l'axe vertical, et **"d"** – qui est un début de segment – correspond à l'axe horizontal. Par suite, l'ensemble des segments de l'Équation 12 correspond à une zone qui apparait sous forme d'un triangle rectangle isocèle, où :

- → L'hypoténuse est parallèle à la deuxième diagonale de la matrice (voir les figures Figure 49 et Figure 50) et,

[3] Un segment qui se situe dans la répétition a une taille plus petite que **"L"**. Et, un segment est considéré périodique s'il contient plus que 2 occurrences.

→ Les deux autres côtés sont parallèles aux axes horizontaux et verticaux et,

→ Il est situé entre les deux lignes : $\mathbf{2 \times O}$ et $\mathbf{L}$ et,

→ Il est situé entre les deux colonnes : $\mathbf{D}$ et $\mathbf{F - 2 \times O}$,

→ L'angle droit est alors situé à la ligne $\mathbf{2 \times O}$ et à la colonne $\mathbf{D}$.

Donc, si on est capable d'extraire les propriétés du triangle alors on sera capable de déduire les détails de la répétition correspondante. Dans la section suivante, on présente une méthode simple qui extrait les triangles, déjà décrit, de la **MATYIN**. Et par suite, on extrait les détails des répétitions.

4.7.2 Détection des triangles

Pour détecter les triangles rectangles isocèles dans **MATYIN**, nous proposons une méthode simple basée sur la détection de leurs côtés à l'aide d'une estimation du gradient appliquée sur l'image représentant la matrice. Nous ne chercherons à repérer que les triangles dont les caractéristiques correspondent à celles décrites dans le paragraphe précédent.

Dans la mesure où le type et l'orientation du triangle sont déjà connus, il suffit de localiser l'angle droit ainsi que la longueur d'un côté adjacent à celui-ci pour être en mesure de localiser le triangle entier. Un point dans **MATYIN** forme l'angle droit d'un triangle recherché si :

→ Il est un point d'intersection de deux segments de même longueur où :

- Le premier est orienté horizontalement vers la droite et,
- Le deuxième est orienté verticalement vers le bas

→ La zone formée par le triangle rectangle dont ces deux segments sont les côtés opposés à l'hypoténuse est remplie par de petites valeurs.

Comme nous n'avons aucune information concernant la position probable ou la longueur de la répétition, le triangle peut être situé à n'importe quelle position dans la matrice. Par conséquent, chaque point de la matrice est une position probable de l'angle droite d'un triangle recherché. Nous proposons d'associer à chaque point de la matrice un vote qui reflète la possibilité qu'il forme l'angle droit d'un triangle. Ce vote dépend de la présence de segments bien marqués à droite et en dessous de ce point et de valeurs faibles remplissant le triangle ainsi défini.

La méthode consiste à appliquer un estimateur de gradient sur la matrice **MATYIN** pour mettre en évidence les variations locales des coefficients. La valeur absolue de l'estimateur doit être d'autant plus élevée en un point que la variation locale est elle-même élevée. Pour ce faire, nous avons choisi l'estimateur de Prewitt.

Ci-dessous sont représentés les gradients des matrices **MATYIN** correspondant aux figures Figure 49 et Figure 50.

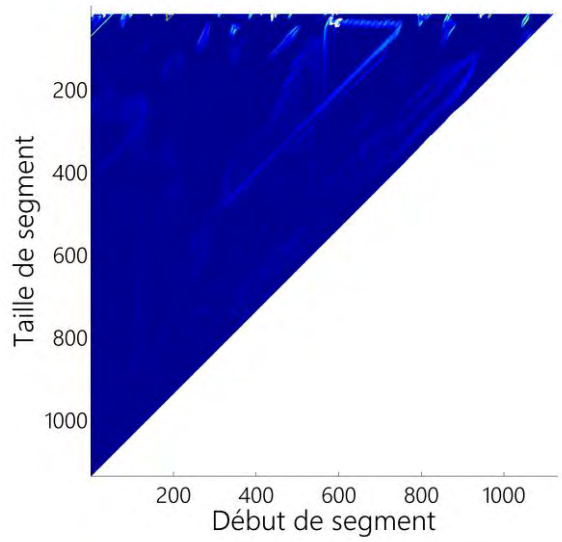


Figure 51: Le gradient correspondant à la vidéo réelle présentée dans la Figure 48. Sa matrice MATYIN est déjà présentée par la Figure 49.

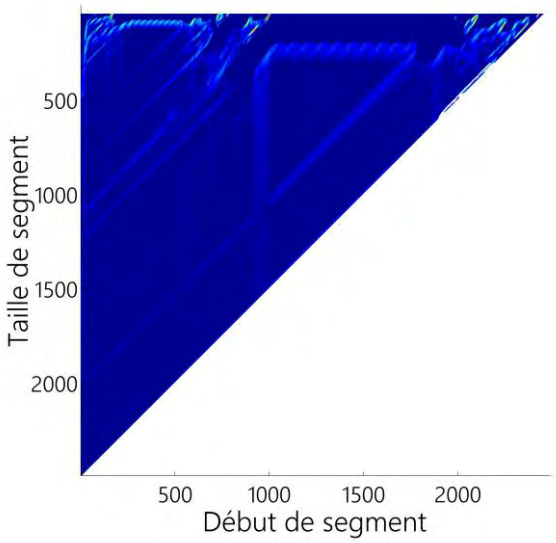


Figure 52 : Le gradient correspondant au signal simulé présenté par la Figure 45, où s. Sa matrice MATYIN est présentée par la Figure 50.

Nous observons comme prévu que les contours horizontaux et verticaux sont associés à des valeurs élevées de l'estimateur de gradient (en valeur absolue). Ceci nous permet de produire un vote pondéré pour chaque point de **MATYIN** en fonction du fait qu'il soit l'angle droit d'un triangle dans la matrice des gradients. La procédure suivante est suivie pour calculer une matrice "**ANGLEDROITE**", où l'entré "**ANGLEDROITE** $_{i,\,j}$" contient le vote correspondant à l'entré "**MATYIN** $_{i,\,j}$".

Soient :

- ✓ M la matrice MATYIN,
- ✓ G la matrice Gradient,
- ✓ t une taille possible de côté d'un triangle,
- ✓ $V_{i,j,t}$ la possibilité que le point (i, j) soit l'angle droit d'un triangle ayant une longueur de côté t.

$$V_{i,j,t} = \frac{\sum_{a=0}^{t}\sum_{b=0}^{t-a} M_{i+a,j+b}}{\frac{1}{2}\times(t+1)^2} \times \frac{\sum_{k=j+1}^{j+t} G_{i,k} + \sum_{k=i+1}^{i+t} G_{k,j}}{2\times t} \qquad \text{Équation 13}$$

Le premier terme de vote, ainsi que nous le proposons de le calculer, est le produit de deux termes. Le premier est la somme des coefficients situés à l'intérieur du triangle rectangle isocèle candidat ; le second est la somme des gradients (en valeur absolue) correspondant à ses côtés de même longueur. Selon l'Équation 13, le vote maximal est celui qui maximise les valeurs collectées par les deux matrices (**M** et **G**). Dans la matrice **GRADIENT**, les valeurs d'intérêt sont les plus élevées. Mais dans la

MATYIN, les valeurs d'intérêt sont les plus basses. Pour assurer un comportement identique de ce second terme avec le premier, nous inversons les valeurs de cette matrice de la manière suivante :

$$MATYIN\_maximisée_{i,j} = \frac{MAX_{théorique} - MATYIN_{i,j}}{MAX_{théorique}}$$

Équation 14

Où **MAX** $_{\text{théorique}}$ est le maximum théorique de **MATYIN**.

Mais selon le principe de **MATYIN**, un maximum théorique n'existe pas, car on ne peut pas limiter la valeur maximale possible de "**my**" dans la courbe YIN (*cf.* 4.3.2 et 4.6). Comme le maximum théorique n'a pas un effet critique sur notre méthode, l'utilisation d'une valeur très grande peut être une solution. Dans notre travail, nous choisissons d'utiliser le maximum de toutes les matrices de notre corpus.

$$MATYIN\_maximisée_{i,j} = \frac{MAX - MATYIN_{i,j}}{MAX}$$

Équation 15

Où **MAX** est la valeur maximale de toutes les **MATYIN** de notre corpus.

En retournant à l'Équation 13, la matrice de vote **V** ainsi obtenue est donc une matrice à trois dimensions où **t** dépend du corpus utilisé et éventuellement de la nature des actions répétées si cette information est disponible a priori. Nous transformons ensuite cette matrice en la matrice **ANGLEDROIT** à deux dimensions en retenant la valeur maximale obtenue sur la dimension **t** en chaque point **i, j**. En d'autres termes, nous cherchons à mettre en évidence, les plus grands triangles possibles.

$$ANGLEDROIT_{i,j} = \max_{t \in T} V_{i,j,t}$$

Équation 16

Au moment d'enregistrer le vote d'un point dans la matrice **ANGLEDROIT**, nous mémorisons également la taille des côtés **t** qui a produit ce vote maximal.

Les figures suivantes montrent les matrices **ANGLEDROIT** des **MATYIN** présentées par les figures Figure 49 et Figure 50.

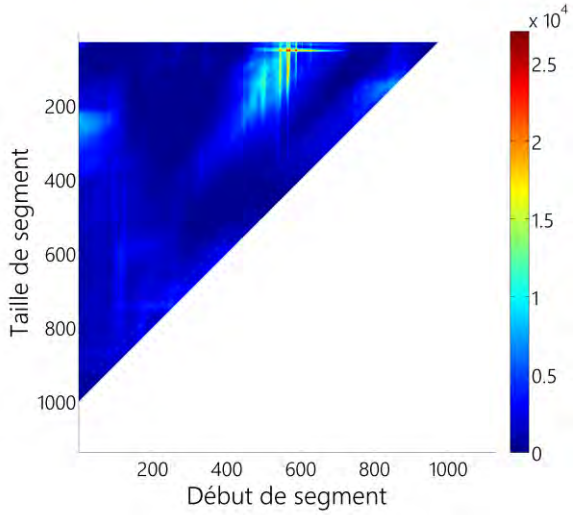


Figure 53: La matrice ANGLEDROIT correspondant à la vidéo réelle représentée par le signal de la Figure 48. Sa matrice MATYIN est déjà présentée par la Figure 49.

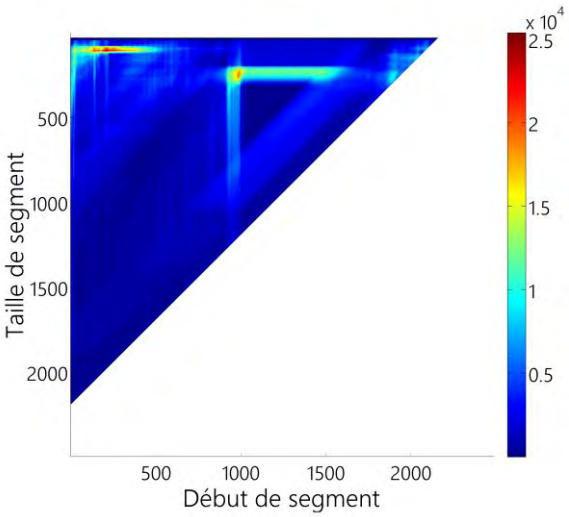


Figure 54 : Le gradient correspondant au signal simulé présenté par la Figure 45, où sa matrice MATYIN est présentée par la Figure 50.

4.7.3 Extraction des triangles

Dans les figures précédentes, nous voyons que les positions des points ayant un vote élevé sont clairement identifiables. Par suite, détecter le point ayant le vote maximal dans **ANGLEDROIT** permet de localiser le début d'une répétition dans **MATYIN**, ainsi que sa durée en fonction de la taille des côtés du triangle.

Cependant, plusieurs répétitions peuvent avoir lieu dans une vidéo donnée, ce qui rend l'extraction d'un vote maximal unique insuffisant pour localiser toutes ces répétitions. Une solution triviale pourrait consister à chercher le deuxième vote maximal puis le troisième et ainsi de suite. Mais d'après la Figure 53, nous remarquons que le deuxième vote maximal correspond au point situé dans un voisinage immédiat de celui du vote maximal. Ainsi, il est fréquent que les plus hauts votes d'une matrice **ANGLEDROIT** correspondent à un ensemble des points situés au voisinage de l'angle droit recherché.

Dans de nombreux cas, ces votes correspondent à des triangles ayant une forte intersection entre eux (ils correspondent généralement à la même solution moyennant de légers décalages liés au calcul du YIN ou à la faiblesse de la résolution dans la vidéo par exemple), Or, la méthode de détection des répétitions repose sur l'hypothèse que deux répétitions n'ont pas d'intersection entre elles dans une vidéo. Par conséquent, deux triangles correspondant à deux répétitions différentes ne sont pas supposés avoir une intersection dans la **MATYIN**. Par suite, nous proposons d'intégrer cette hypothèse dans la méthode d'extraction des votes maximaux pour repérer un angle droit unique associé à chaque triangle. L'algorithme est le suivant :

Algorithme :

```
Boucle infinie
Début
    Trouver le point ayant le vote maximal dans ANGLEDROIT ;
    Si le vote maximal est égal à zéro :
    Début
        Quitter la boucle ;
    Fin
    Si le triangle correspondant ne présente pas une intersection avec
un  triangle déjà extrait :
    Début
        Sauvegarder ce point comme étant l'angle droit d'un triangle ;
        Sauvegarder la taille correspondant aux côtés ;
        Si on ne veut plus extraire de nouveaux triangles :
        Début
            Quitter la boucle ;
        Fin
    Fin
    Mettre ce point à zéro ;
Fin
```

Cet algorithme suppose que nous disposions d'une information sur le nombre de triangles à extraire. Le fait d'extraire les triangles de chaque point ayant un vote non nul, conduit à extraire un nombre indéterminé de répétitions supposées, ce qui n'a pas de sens. Donc, un point d'arrêt de cet algorithme doit être défini :

→ Définir un seuil de vote d'une manière que l'algorithme s'arrête une fois que le vote maximal soit plus petit que le seuil défini. Cependant, la définition d'un seuil optimal de manière universelle est un grand problème en raison de la diversité des variables à prendre en considération.

→ Atteindre un nombre défini de triangles détectés : l'algorithme s'arrête lorsqu'un nombre prédéfini de triangles distincts a été extrait.

La première solution serait idéale s'il apparaissait possible de trouver une valeur de seuil de manière automatique et adaptative. Cependant, il est impossible de calculer un seuil valide pour toutes les vidéos, car les valeurs calculées dans les matrices dépendent fortement du type de contenu. Par exemple, le cas d'une vidéo qui contient une première répétition à des occurrences très similaires et d'autres répétitions ayant des occurrences moins similaires entre elles. Dans la matrice **ANGLEDROIT**, le vote de l'angle droit de la première répétition est très élevé par rapport aux votes des autres répétitions. Dans ce cas, on doit choisir un seuil bas pour inclure tous les angles correspondants aux répétitions. Par contre, dans le cas d'une vidéo où toutes les répétitions ont à peu près le même niveau de similarité entre les occurrences, un petit seuil va inclure des points qui ne correspondent pas à des répétitions. Dans ce cas le seuil doit être plus élevé que celui du premier cas.

À l'inverse, le cadre de notre travail de recherche suppose que nous disposions d'outils d'analyse textuels permettant d'identifier (entre autre) des instructions simples,

des instructions conditionnelles, et des boucles. Donc, le nombre des répétitions à localiser dans la vidéo est supposé fourni par les outils textuels, ce qui nous permet d'opter pour la deuxième solution.

Cependant, le fait d'extraire un nombre de triangles exactement égal au nombre des répétitions cherchées n'est acceptable qu'en supposant que l'outil proposé dans ce chapitre est complétement fiable, ce qui n'est pas le cas. Si nous choisissons d'extraire uniquement un nombre de triangles égal au nombre des boucles dans le texte, nous risquons de rater des triangles corrects (correspondant à des répétitions mais de valeur de vote inférieure à des cas erronés). Dans la section suivante (section 4.8.3.4), nous étudierons la relation entre le nombre des triangles extraits et le taux de répétitions sélectionnées correctement.

4.7.4 Extraction des paramètres des répétitions

Une fois les triangles trouvés, nous sommes capables d'en tirer des informations sur les répétitions correspondantes. L'examen des segments définis par l'Équation 12, ainsi que leurs significations au début de cette section conduit aux observations suivantes :

Étant données :

→ La position de l'angle droit (**Ligne**, **Colonne**) et,

→ la taille des côtés : **Coté**

Début = Colonne

→ Selon l'Équation 12, pour chaque taille "**l**" de segment, le premier segment dans l'ensemble "**Seg**" commence au début de la répétition "**D**". Les coefficients correspondants dans **MATYIN** forment le côté vertical du triangle "**colonne**".

Durée = Ligne + Coté

→ C'est la taille du plus grand segment défini dans l'ensemble "**Seg**". Graphiquement, cette valeur correspond au numéro de ligne de l'angle situé au-dessous de l'angle droit.

Fin = Début + Durée

→ La fin de la répétition est située à une distance "**Durée**" après son début.

Durée d'une occurrence = Ligne / 2

→ Les plus petits segments dans l'ensemble "**Seg**" comportent deux occurrences "**2xO**". Graphiquement, les coefficients de ces segments forment le côté horizontal du triangle. Leur taille est égale à "**Ligne**". De ce fait, la taille "**O**" d'une occurrence est la moitié de "**Ligne**".

Nombre d'occurrences = Durée / (Durée d'une occurrence)

→ C'est la durée totale divisée sur la durée d'une occurrence.

	MATYIN de la Figure 49	MATYIN de la Figure 50	
		1ère répétition	2ème répétition
Début	563	208	989
Durée	212	414	988
Fin	775	622	1977
Nombre des occurrences	8	8	8
Durée d'occurrence	26	51	124

Tableau 8 : Résultats correspondants aux exemples présentés depuis le début de ce chapitre.

4.8 Expérimentations et évaluation

Nous cherchons maintenant à évaluer l'efficacité de la méthode que nous venons de proposer et sa capacité à localiser les répétitions dans les vidéos. Cette évaluation s'effectue en deux étapes : la préparation des corpus adaptés à cette évaluation et ensuite l'évaluation à proprement parler.

4.8.1 Corpus et Résultats

Pour vérifier que la méthode proposée se comporte bien, elle doit être testée sur une grande diversité de situations possibles. Pour cela, nous avons considéré deux types de contenus : des contenus simulés et des vidéos réelles.

4.8.1.1 Contenus simulés

Le but de tester la méthode sur des contenus simulés est d'étudier son comportement dans des cas contrôlés où l'impact d'un facteur donné peut être étudié séparément. Nous vérifions ensuite que le passage vers des contenus réels, pour lesquels ces facteurs ne peuvent plus être contrôlés, confirme les premiers résultats.

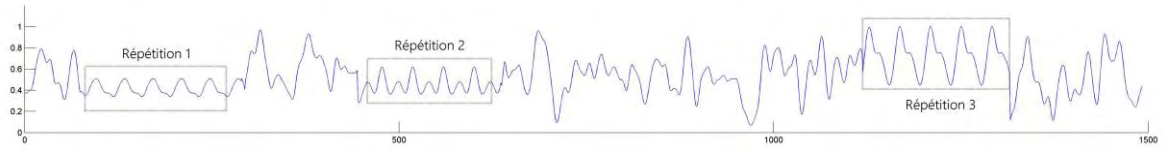
Comme on l'a déjà mentionné, la méthode nécessite en entrée un signal sensible aux répétitions qu'il faut détecter. Dans un premier temps, nous souhaitons évaluer la performance de la méthode indépendamment de ce signal. Le choix du signal sera étudié ultérieurement. Pour cela, les contenus simulés sont des signaux qui contiennent une ou plusieurs répétitions situées à des emplacements précis et notre but est alors d'observer si la méthode permet de le repérer et d'en extraire les paramètres avec une précision suffisante.

On a étudié l'effet de plusieurs facteurs sur les résultats de la méthode en produisant chaque fois un signal avec au moins une répétition, et affecté par différents facteurs.

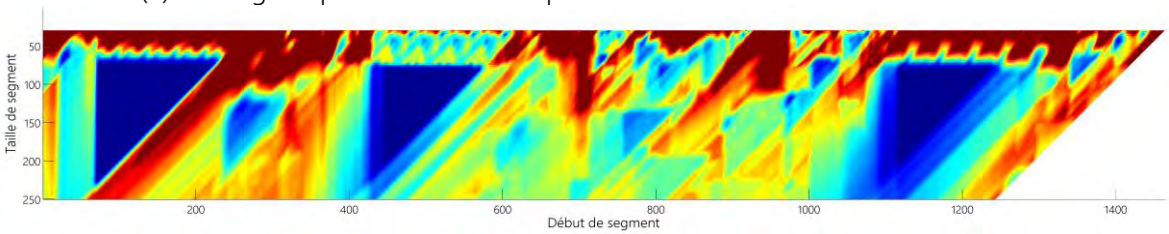
4.8.1.1.1 Forme simple :

Les signaux sont produits en deux étapes : construire des signaux à contenu aléatoire et puis ajouter des segments périodiques pour simuler les répétitions. Au début, nous utilisons la fonction "**randn**" définie sous MATLAB pour construire des séquences des valeurs aléatoires tirées selon une distribution normale. Afin d'avoir des signaux similaires aux signaux des vidéos réelles, nous choisissons de lisser les séquences ainsi produites. Ensuite, nous construisons à part des séquences périodiques pour les insérer à des positions aléatoires dans les signaux. La procédure suivante est suivie pour construire et insérer une répétition dans un signal donné :

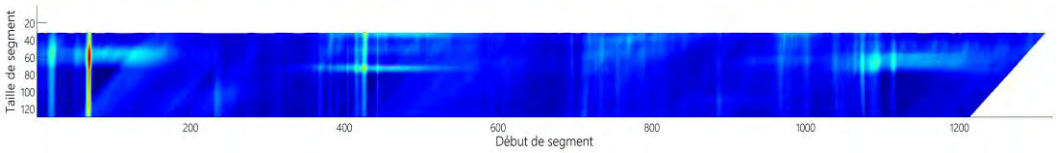
1) *Construire un signal (S) de taille aléatoire et formé des valeurs aléatoires*
2) *Lisser ce signal en utilisant un filtre médian avec une taille de fenêtre qui exige un signal lisse*
3) *Construire un petit signal (s) de la même manière que celle utilisé à l'étape 1 (sa taille est entre 6 et 42),*
4) *Construire un signal (R) qui consiste en la concaténation de plusieurs copies (entre 5 et 10) du signal (s),*
5) *Lisser le signal (R) en utilisant le même filtre médian de l'étape 2,*
6) *Choisir une position aléatoire dans le signal (S), et puis insérer (R).*



(a) : Un signal qui contient trois répétitions à côté des autres contenus aléatoires.



(b) : La matrice MATYIN correspondante au signal ci-dessus (a).



(c) : La matrice ANGLEDROIT correspondant à (b).

Figure 55 : Exemple d'un signal simulé.

Nous remarquons dans l'exemple ci-dessus que les répétitions sont bien représentées par des triangles de faibles coefficients dans la **MATYIN**, et que les angles droits reçoivent des votes élevés dans **l'ANGLEDROIT**.

Nous rappelons ici que nous n'évaluons pas la performance de notre méthode sur un corpus simulé qui prend en considération tous les cas possibles (cette méthode est originalement orientée vers des contenus visuels et non pas des signaux), mais nous présentons des exemples démonstratifs et illustratifs qui montrent la démarche de la méthode et le type des résultats attendus.

Le corpus simulé est formé de 10 signaux aléatoires dont chacun contient cinq répétitions à côté des autres contenus aléatoires (similaire à Figure 55 – (a)), cela donne la possibilité de détecter 50 répétitions de taille, période et fréquence différentes. Le tableau suivant présente les détails des répétitions et des signaux dans ce corpus :

	Description	Valeur
Taille moyenne des signaux	Moyenne (tailles des signaux)	2120 trames
Taille moyenne des répétitions	Moyenne (durées des répétitions sélectionnées automatiquement)	140.9 trames
Taille minimale des répétitions	La plus petite taille d'une répétition	60 trames
Taille maximale des répétitions	La plus grande taille d'une répétition	210 trames
Période moyenne	Moyenne (périodes des répétitions)	20.6 trames
Période minimale	La période la plus courte	6 trames
Période maximale	La période la plus longue	42 trames
Fréquence moyenne	Moyenne (fréquences des répétitions)	7.2
Fréquence minimale	La fréquence la plus petite	5
Fréquence maximale	La fréquence la plus grande	10

Tableau 9 : Détails des répétitions dans le corpus.

La détection automatique de ces répétitions produit un taux d'erreur négligeable (voir le tableau ci-dessous)

	Description	Valeur
Erreur moyenne de localisation des débuts	Moyenne (Nombre des trames entre la position de chaque début sélectionné automatiquement et sa position correcte)	6 trames
Erreur moyenne de localisation des fins	Moyenne (Nombre des trames entre la position de chaque fin sélectionnée automatiquement et sa position correcte)	0.4 trames

Erreur moyenne de période	Moyenne (périodes déduites automatiquement)	1.6 trames
Erreur moyenne de fréquence	Moyenne (fréquences déduites automatiquement)	0.1

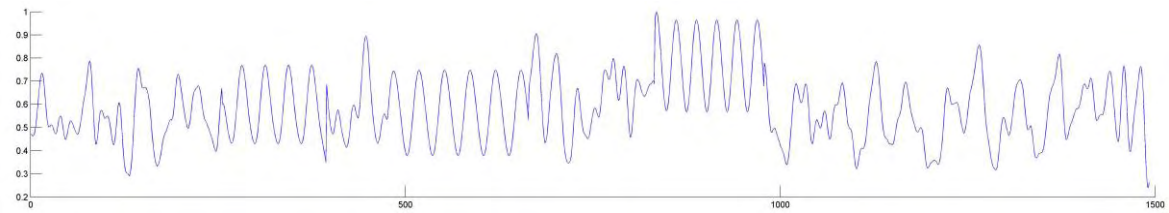
Tableau 10 : Les résultats en bref.

4.8.1.1.2 Bruit

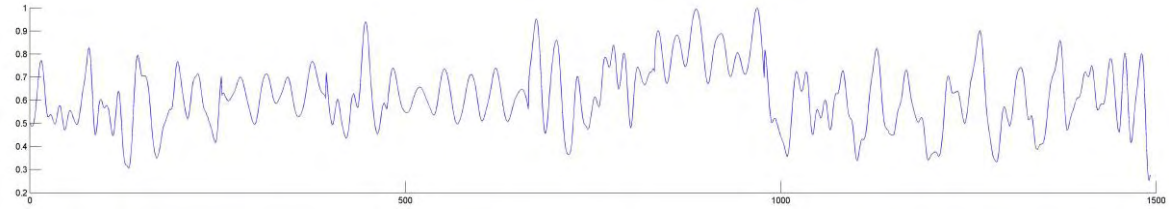
Nous cherchons maintenant à étudier l'impact d'une variation de l'exécution d'un geste au cours de sa répétition sur la capacité à détecter cette répétition. Nous représenterons cette variation par un "bruit". Ce bruit peut simuler aussi des effets liés à la mauvaise qualité de vidéo ou au manque de précision de la méthode de production du signal.

Le bruit consiste ici en l'insertion automatique de valeurs aléatoires à des positions aléatoires dans une répétition donnée. Par exemple, nous désignons par un niveau de bruit 60% dans un signal donné, le fait d'insérer dans chaque répétition de ce signal un nombre des valeurs aléatoires égal à 60% de sa taille, d'où la nouvelle procédure de création des signaux :

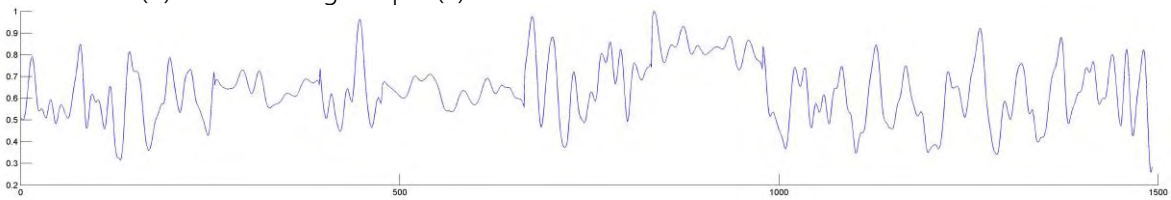
1) *Construire un signal **(S)** de taille aléatoire et formé des valeurs aléatoires*
2) *Lisser ce signal en utilisant un filtre médian avec une taille de fenêtre qui exige un signal lisse*
3) *Construire un petit signal **(s)** de la même manière que celle utilisé à l'étape 1 (sa taille est entre 6 et 42),*
4) *Construire un signal **(R)** qui consiste à la concaténation de plusieurs copies (entre 5 et 10) du signal **(s)**,*
5) *Pour un niveau de bruit "**N**", remplacer "**k**" entrés différentes (sélectionnées aléatoirement) dans **(R)** par des valeurs aléatoires comprises entre le minimum et le maximum des valeurs dans le signal.*
   *Où, **k = N * taille_de_(R) /100**.*
6) *Lisser le signal **(R)** en utilisant le même filtre médian de l'étape 2,*
7) *Choisir une position aléatoire dans le signal **(S)**, et puis insérer **(R)**.*



(a) : Un signal qui contient trois répétitions non bruitées à côté des autres contenus aléatoires.



(b) : le même signal que (a) mais avec un niveau de bruit de 40%.



(c) : le même signal de (a) mais avec un niveau de bruit 80%.

Figure 56 : Exemple d'un signal et deux versions plus ou moins bruitées.

De même que dans la section précédente, nous construisons 10 signaux dont chacun contient 5 répétitions différentes (Tableau 11). Ensuite, nous construisons 9 versions de chaque signal de sorte que chaque version a un niveau différent de bruit (10%, 20%, ..., 90%). Afin de visualiser l'effet du bruit sur les résultats de notre méthode, nous calculons les paramètres suivants pour chaque niveau de bruit :

- → Erreur moyenne de localisation des débuts
- → Erreur moyenne de localisation des fins
- → Erreur moyenne de période
- → Erreur moyenne de fréquence

	Description	Valeur
Taille moyenne des signaux	Moyenne (tailles des signaux)	2135 trames
Taille moyenne des répétitions	Moyenne (durées des répétitions sélectionnées automatiquement)	190.5 trames
Taille minimale des répétitions	La plus petite taille d'une répétition	60 trames
Taille maximale des répétitions	La plus grande taille d'une répétition	210 trames
Période moyenne	Moyenne (périodes des répétitions)	27.6 trames
Période minimale	Le période la plus courte	6 trames
Période maximale	La période le plus longue	42 trames
Fréquence moyenne	Moyenne (fréquences des répétitions)	7.2
Fréquence minimale	La fréquence la plus petite	5
Fréquence maximale	La fréquence la plus grande	10

Tableau 11 : Information du corpus utilisé.

Niveau de bruit	Erreur moyenne de localisation des débuts (en trame)	Erreur moyenne de localisation des fins (en trame)	Erreur moyenne de période (en trame)	Erreur moyenne de fréquence
0 %	14.72	27.74	7.9	1.6
10 %	19.42	36.46	11.77	2.74
20 %	28.48	43	11.95	3.42
30 %	34.76	57.5	11.94	4.84
40 %	33.10	72.08	10.33	4.96
50 %	45.16	67	16.59	4.08
60 %	61.68	71.78	18.93	5.86
70 %	49.52	81.26	21.48	9.3
80 %	52.56	89.06	18.49	5.28
90 %	72.24	80.7	21.74	8.44

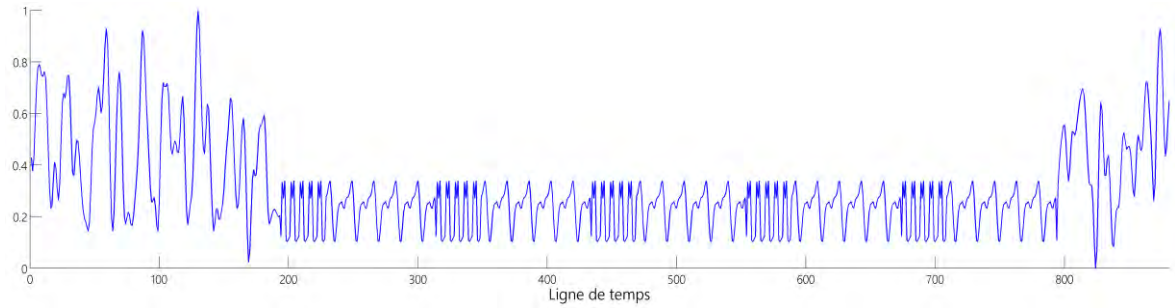
Tableau 12 : Erreur de localisation pour chaque niveau de bruit.

Nous remarquons que le bruit affecte la précision de la méthode en fonction de son niveau. Le tableau ci-dessus montre que l'erreur de localisation des répétitions peut être de 3 à 4 secondes. La période et la fréquence sont également fortement affectées par le bruit.

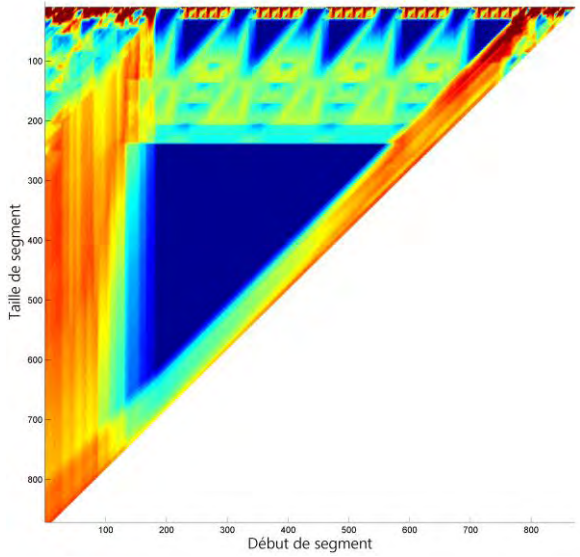
4.8.1.1.3 Répétitions imbriquées

Dans cette section, nous montrons le comportement de notre méthode dans le cas de répétitions imbriquées. Ce cas correspond dans le contenu textuel aux boucles inclues dans une autre boucle.

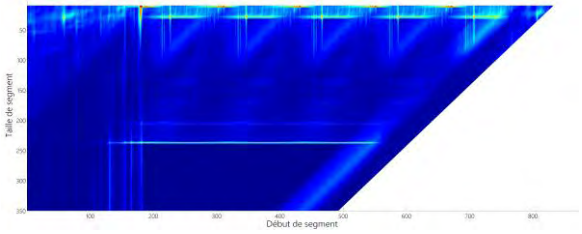
La figure suivante montre un signal qui contient une répétition globale qui consiste à deux "sous-répétitions", ainsi que les matrices **MATYIN** et **ANGLEDROIT** correspondantes :



(a) Un signal simulé qui contient des répétitions imbriquées. La répétition globale consiste en la répétition consécutive de deux sous-répétitions. Taille du signal=880, taille de la répétition globale=600, taille de la 1ère (resp. 2ème) sous-répétition=25 (resp. 60).



(b) La MATYIN du signal (a).



(c) La matrice ANGLEDROIT correspondante à (b).

Figure 57 : Exemple d'une répétition imbriquée.

Comme nous pouvons nous y attendre, chaque "sous-répétition" est représentée par un triangle. Donc la détection de ce type de répétition ne diffère pas beaucoup de celle des répétitions simples, pour lesquelles notre méthode fonctionne bien.

Mais, nous remarquons l'apparition d'un nouvel triangle au-dessous des autres. Ce triangle qui correspond à la répétition globale, permet d'en extraire ses informations (début et fin). Ces informations peuvent aussi être déduites à partir de l'ensemble des triangles en haut (ceux des sous répétition) : son début = (début de la première sous répétition), et sa fin = (fin de la dernière sous répétition). Donc, le triangle au-dessous délimite la répétition globale et les autres triangles délimitent les sous-répétitions à l'intérieur de la globale.

Plus loin, nous testerons cette méthode sur une vidéo réelle qui contient des répétitions imbriquées. Nous verrons que les matrices **MATYIN** obtenues sur un signal réel sont assez différentes de celle d'un signal simulé (Figure 57 – b), en particulier pour

ce qui concerne le triangle de la répétition globale. Les informations de cette répétition devront être déduites à partir des triangles des sous répétitions.

4.8.1.2 Vidéos réelles

Pour évaluer la méthode sur des enregistrements réels, des corpus adaptés doivent être préparés. Nous avons choisi de contraindre la nature des vidéos tournées à cette fin, avec l'idée que l'extension à des cas plus généraux devrait faire l'objet de travaux et de propositions complémentaires.

Ces contraintes sont les suivantes :

- → La caméra de capture de la scène est unique et stationnaire,
- → Tout mouvement correspond forcement à celui de l'opérateur,
- → L'opérateur exécute consécutivement un ensemble d'actions correspondant à une procédure décrite de manière textuelle.

Dans la section de l'état de l'art, nous avons recensé des méthodes pour détecter les mouvements périodiques dans une vidéo ((17), (18), (7) et (14)). Nous avons mentionné que ces méthodes peuvent être adaptées pour localiser les répétitions, ainsi que leurs paramètres. Par contre, les corpus utilisés dans ces travaux durant l'évaluation des méthodes ne sont pas disponibles ou bien ne satisfont pas nos hypothèses pour deux raisons :

- → Les vidéos sont enregistrées par une caméra mobile,
- → Les vidéos ne contiennent qu'une répétition qui commence au début de la vidéo et se termine à la fin annulant tout intérêt pour une détection et une localisation automatique.

En raison de l'originalité du sujet abordé dans ce chapitre, l'évaluation de la méthode sur une grande base de données référencée est pratiquement impossible. Nous avons donc créé notre propre corpus comprenant des vidéos variées dans une perspective de test et de vérification. Toutefois, la taille de ce corpus ne sera pas totalement suffisante pour en tirer des conclusions générales sur la performance de la méthode.

Les vidéos de notre corpus sont toutes enregistrées par la même caméra semi-professionnelle assurant la production d'enregistrements de qualité élevée. La caméra est fixée sur un trépied afin d'éviter tout tremblement durant l'enregistrement. D'autre part, les scènes sont enregistrées dans différents environnements : arrière-plans (bureau, appartement, toit) et types d'éclairage (fluorescence, incandescence, LED, soleil). Dans chaque scène, un opérateur exerce plusieurs actions individuelles comportant une ou plusieurs répétitions.

Nombre des vidéos	14 vidéos
Taille totale des vidéos	27720 trames
Taille totale des répétitions	11837 trames
Nombre des répétitions	27 répétitions
Taille moyenne des répétitions	438.407 trames

Tableau 13 : Détails du corpus utilisé.

4.8.1.2.1 Sans répétition

Nous examinons dans un premier temps le comportement du système quand la vidéo ne contient aucune répétition pour étudier la capacité du système à produire des faux positifs. L'exemple suivant présente le signal et la matrice MATYIN correspondants à une vidéo où l'opérateur exécute trois actions individuelle différentes sans aucune répétition.

Figure 58: Signal correspondant à une vidéo qui ne contient pas de répétition.

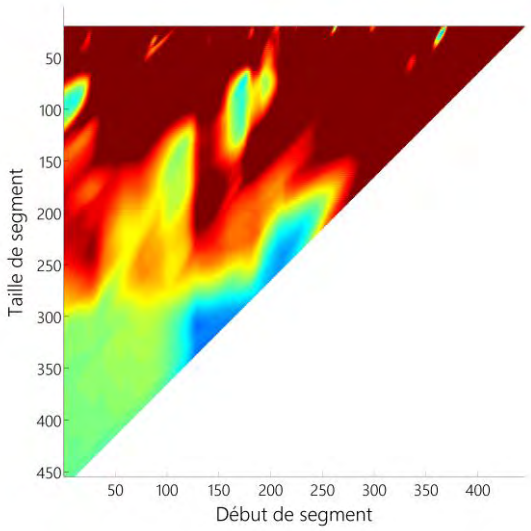


Figure 59: La matrice MATYIN correspondant au signal de la Figure 58.

On remarque clairement que la matrice ne contient aucun triangle. On peut toujours trouver un maximum dans la matrice **ANGLEDROIT**, et par suite nous pouvons déduire l'existence d'une répétition. Durant l'expérimentation, nous remarquons que les votes qui correspondent aux répétitions sont très élevés par rapport aux votes des autres points. Dans le tableau suivant, nous présentons un ensemble des votes

maximaux extraits des matrices **ANGLEDROIT** qui correspondent à des vidéos sans répétition et à d'autres avec répétition. Ce tableau montre que les votes des points qui correspondent aux répétitions (Tableau 14 – colonne 4) sont clairement plus élevés que ceux des autres points (Tableau 14 – colonne 2).

Vidéos sans répétition	Vote maximal	Vidéos avec répétition	Vote maximal
Vidéo 1	287.3941	**Vidéo 7**	10820.0506
Vidéo 2	244.4395	**Vidéo 8**	5421.63
Vidéo 3	348.0798	**Vidéo 9**	7184.461
Vidéo 4	298.4653	**Vidéo 10**	10545.859
Vidéo 5	279.5422	**Vidéo 11**	8381.1024
Vidéo 6	393.2515	**Vidéo 12**	4118.9544

Tableau 14 : deux ensembles des votes maximaux extraits des matrices ANGLEDROIT. Le premier (vidéo 1 → 6) correspondent aux vidéos sans répétition et le deuxième (vidéo 7 → 12) correspondent aux vidéos avec répétition.

4.8.1.2.2 Vidéo à une ou plusieurs répétitions

Deux types de répétition sont considérés dans cette section : "répétition périodique d'une action" et "répétition d'actions similaires".

1) Répétition périodique d'une action

Un exemple de ce type des répétitions est l'exécution d'exercices sportifs (ex. exercices de musculation, de gym, d'entraînement répétés en séries). On observe sur ces enregistrements des occurrences relativement similaires où les légères variations sont dues à l'incapacité d'un humain à répéter la même action d'une manière strictement identique.

Nombre des vidéos	13 vidéos
Tailles des vidéos	25 → 122 secondes
Durée totale	515 secondes
Vidéo à une répétition	8 vidéos
Vidéo à deux répétitions	3 vidéos
Vidéo à trois répétitions	2 vidéos
Nombre d'occurrences	3 → 10 occurrences

Tableau 15 : Les enregistrements qui contiennent des répétitions périodiques.

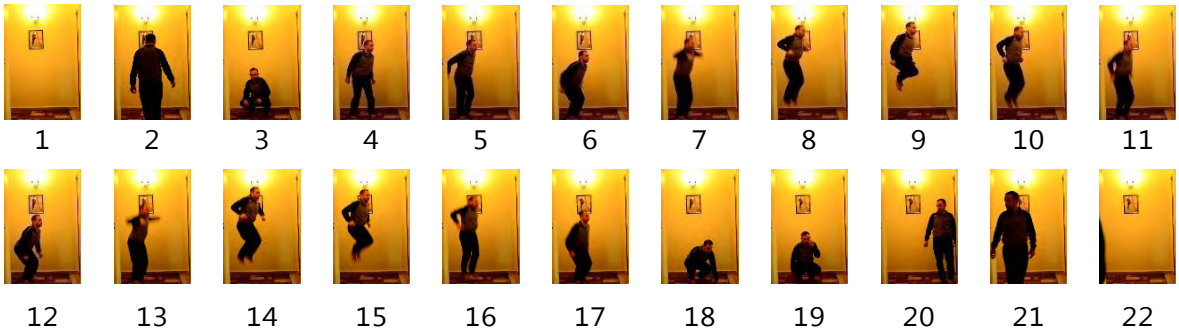


Figure 60: échantillon d'une vidéo (Vidéo 6) dans laquelle un opérateur effectue un exercice sportif (sauter).

2) *Répétition d'actions similaires*

Ce type de répétition consiste à effectuer consécutivement des actions similaires en termes d'effets mais dont l'exécution peut prendre des formes variées. Ce cas est plus complexe que le précédent car il est difficile de trouver une caractérisation générique qui produise un signal numérique où les occurrences apparaitraient de manière similaire. Dans nos expérimentations, nous continuerons à utiliser la courbe de l'intensité moyenne, ce qui affectera naturellement la qualité des résultats dans ce cas de figure.

Les vidéos contenant une répétition de ce type ont les caractéristiques suivantes :

Nombre des vidéos	10 vidéos
Tailles des vidéos	30 → 116 secondes
Durée totale	645 secondes
Nombre d'occurrences	3 → 10 occurrences

Tableau 16 : Enregistrements qui contiennent des répétitions d'actions similaires.

Figure 61: échantillons d'une vidéo (roof 1) dans laquelle l'opérateur effectue une répétition du deuxième type (déplacement). On remarque que les actions de déplacements des objets varient en termes de vitesse, d'objet déplacé, et de mouvement de l'opérateur à chaque déplacement, ce qui différencie cette répétition de celles du premier type.

4.8.1.2.3 Répétitions imbriquées

Une répétition imbriquée consiste à une répétition dont les occurrences sont elles-mêmes des répétitions. Par exemple :

Répéter 3 fois :
Début
    Ajouter 4 cuillères de sucre ;
    Mélanger le contenu du bol avec une cuillère ;
Fin

Comme la matrice **MATYIN** représente chaque répétition par un triangle, alors la matrice correspondant à cet exemple présente 7 triangles : 6 triangles pour les six sous répétitions (1-2-1-2-1-2) et un pour la répétition globale (*cf.* Figure 57).

La détection des répétitions imbriquées ne diffère pas beaucoup de la détection des répétitions simples, car il suffit d'extraire les triangles dans la matrice MATYIN et de déduire les informations des répétitions. Mais nous présentons ce cas indépendamment pour montrer que le nouveau triangle (le 7ème) peut être utile pour affiner les informations extraites des triangles des sous répétitions.

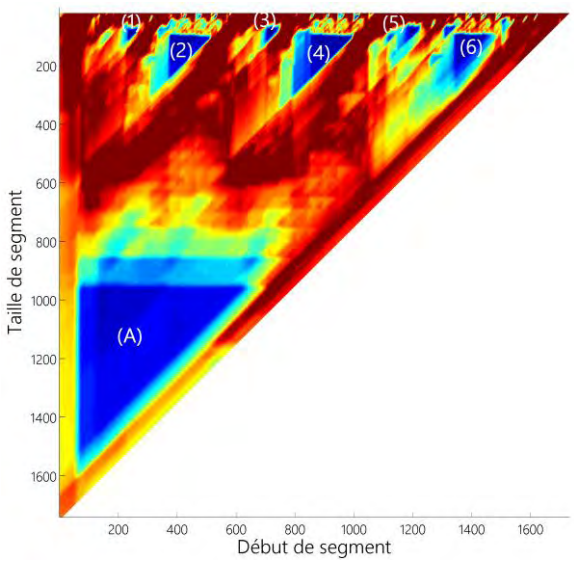


Figure 62 : Matrice MATYIN d'une vidéo qui contient des répétitions imbriquées. Le triangle (A) correspond à la répétition globale et les autres triangles (1→6) correspondent aux sous répétitions ayant lieu dans la répétition globale.

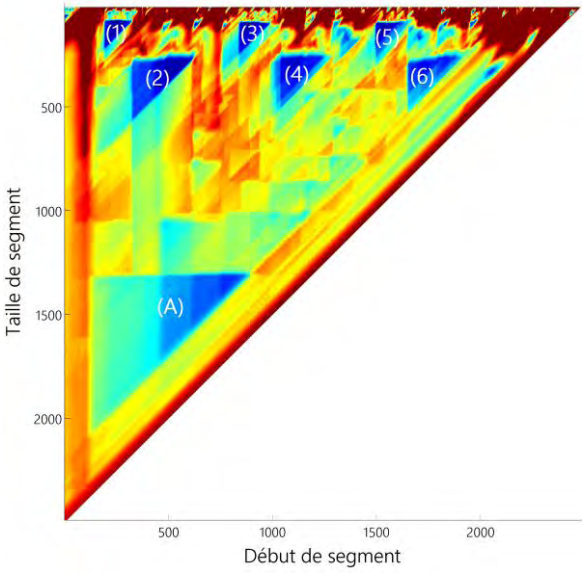


Figure 63 : MATYIN d'une vidéo qui contient une répétition imbriquée. On remarque que le triangle de la répétition globale n'est pas bien placé et est réduit (il doit être à la même colonne que celui de la sous répétition 1).

Dans la Figure 62, nous remarquons le triangle (A) qui correspond à la répétition globale. Ce triangle qui fournit des informations sur cette répétition (début et fin) n'est pas toujours remarquable dans les cas des vidéos réelles (Figure 63), en raison de la variabilité dans une répétition qui augmente potentiellement avec sa durée. Dans le cas

où ce triangle n'apparait pas, nous déduisons les limites de la répétition globale à partir des triangles des sous répétitions (début=début de la première sous répétition, et fin=fin de la dernière sous répétition).

Dans la suite de ce travail, nous choisissons de ne pas traiter les répétitions imbriquées d'une manière spéciale, mais nous allons considérer les sous répétitions comme étant des répétitions indépendantes.

4.8.2 Les résultats sur une vidéo réelle

On présente dans cette section un exemple réel simple pour étudier l'efficacité de la méthode proposée sans avoir d'information a priori sur le contenu. Étant donnée une recette de cuisine, la vidéo montre une personne qui suit les instructions. Une recette comprend parfois des répétitions d'instruction jusqu'à une limite donnée (condition ou nombre d'occurrence). Notre but est d'extraire les répétitions effectuées dans la vidéo ainsi que leurs paramètres.

Dans cet exemple, la recette est la suivante :

- → **Préparer un bol d'eau et un récipient de sucre,**
- → **Ajouter 10 cuillères de sucre,**
- → **Libérer le plan de travail en éloignant le récipient de sucre,**
- → **Bien mélanger le contenu du bol jusqu'à que le sucre soit complètement dissout,**
- → **Préparer la farine, le crème et la levure,**
- → **Ajouter les trois ingrédients consécutivement.**

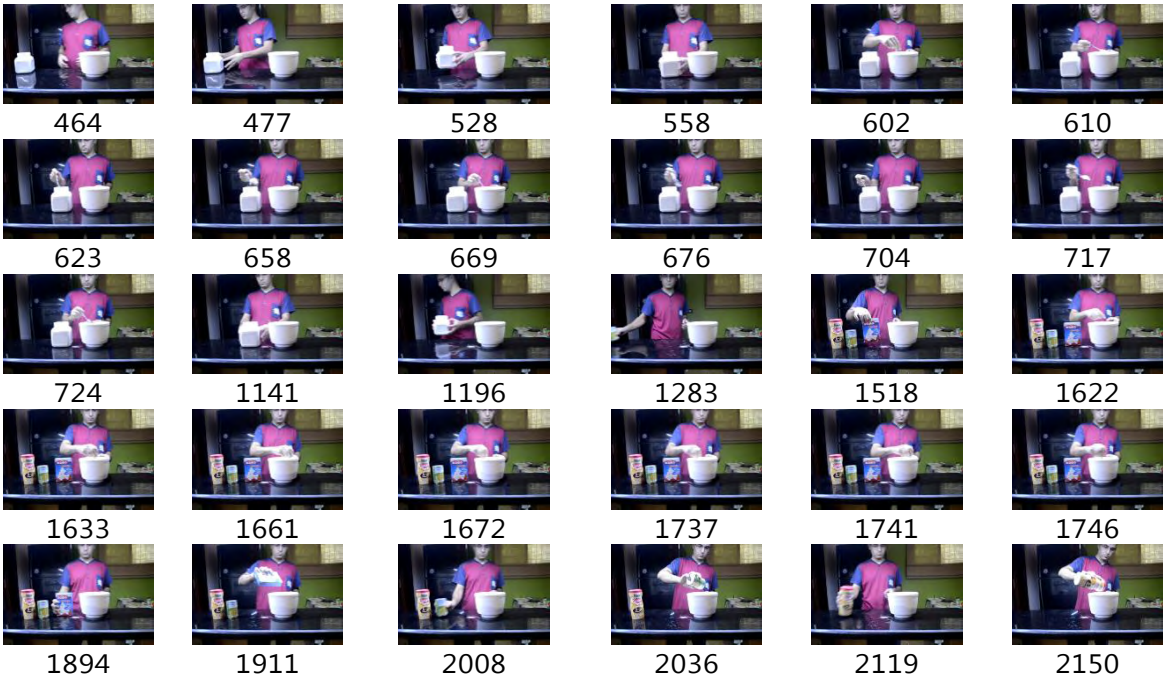


Figure 64: échantillons d'une vidéo (Cuisine 4) qui décrit la préparation d'une recette à la cuisine. Dans cette vidéo, il existe deux répétitions distantes en boucle ("Ajout de sucre" et "mélanger le contenu").

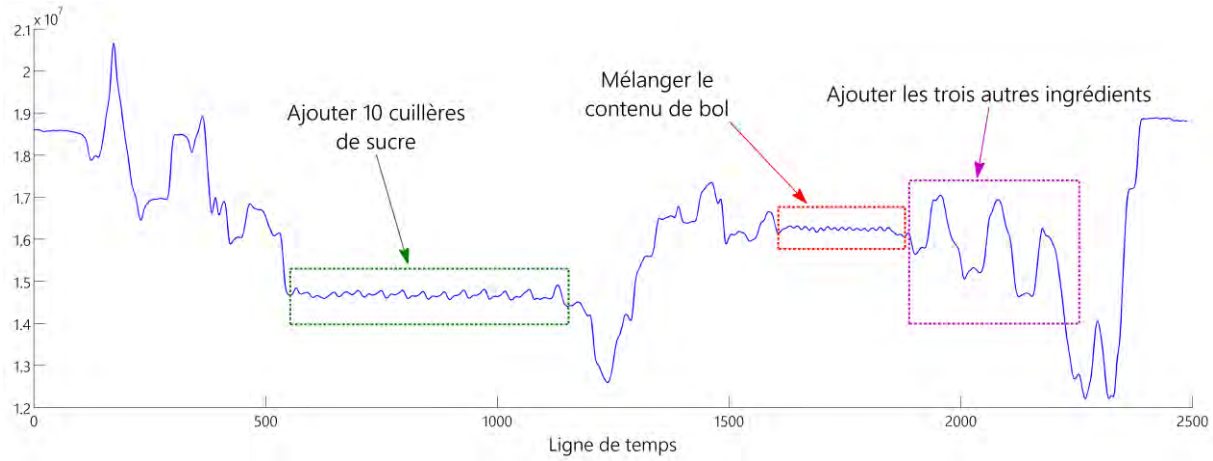


Figure 65: Le signal correspondant à la vidéo de la Figure 64.

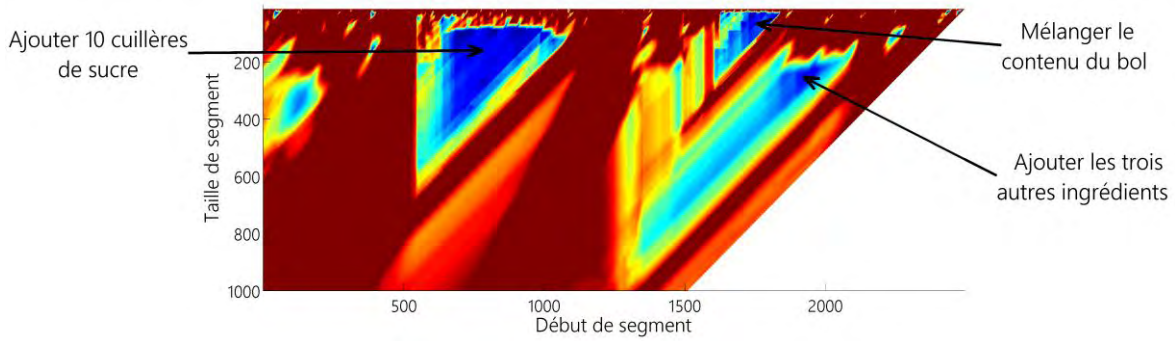


Figure 66: La matrice MATYIN de la vidéo à la Figure 64.

Cette vidéo contient trois répétitions : deux du premier type et une du deuxième type. Dans ce cas de figure, la mesure numérique caractérisant le contenu (i.e. l'intensité moyenne) a à peu près bien représenté les deux types de répétitions puisqu'elles apparaissent sous forme de triangles de petites valeurs dans la matrice **MATYIN**. D'autre part, on remarque que le troisième triangle n'est pas aussi net que les deux autres, car la troisième répétition est du deuxième type, et elle induit d'importantes variations sur la courbe d'intensité. Malgré ça, le triangle correspondant apparait d'une manière acceptable dans la matrice. L'extraction des trois principaux triangles (*cf.* "Extraction des triangles") conduit à produire les valeurs suivantes des paramètres des répétitions (Tableau 17).

	1ère répétition			
	Début	**Fin**	**Nombre des occurrences**	**Durée d'occurrence**
Vérité terrain	635	1090	10	45.5
Résultats de la méthode	634	1074	10	44
	2ème répétition			
	Début	**Fin**	**Nombre des occurrences**	**Durée d'occurrence**
Vérité terrain	1630	1860	14	16.4
Résultats de la méthode	1623	1799	11	16
	3ème répétition			
	Début	**Fin**	**Nombre des occurrences**	**Durée d'occurrence**
Vérité terrain	1885	2210	3	108
Résultats de la méthode	1840	2121	3	93.6

Tableau 17: Comparaison des paramètres des répétitions détectées automatiquement et de celles segmentées à la main. Les valeurs des paramètres des deux premières répétitions extraites automatiquement sont très proches de la vérité terrain. Celles de la troisième répétition qui correspond au deuxième type de répétition, sont moins bonnes mais elles restent relativement acceptables.

4.8.3 Évaluation

Malgré sa faible représentativité, nous allons utiliser notre base d'enregistrements pour évaluer notre méthode afin de donner une idée de la qualité des résultats – qui ne sera pas, en tout état de cause, une information généralisable.

Nous proposons une évaluation en deux étapes :

→ la première consiste à calculer un taux d'erreur global de notre système,

→ la deuxième à comparer ce résultat aux taux d'erreur de deux autres systèmes : un système de type "Oracle" et un système aléatoire pour situer notre méthode par rapport respectivement au meilleur et au pire des cas.

4.8.3.1 Première étape

On désigne par "informations de la vérité terrain", les limites des segments qui entourent une répétition. Ces limites sont sélectionnées manuellement afin de former une référence avec laquelle on peut comparer les limites sélectionnées automatiquement. Cette comparaison nous permet de calculer un taux d'erreur.

Dans le système global de synchronisation, nous choisissons d'extraire de la vidéo un nombre de répétitions égal au nombre de boucles dans le contenu textuel (nous discuterons de cette idée plus tard dans le paragraphe "4.8.3.4" et suivants). Ensuite, nous mettons en correspondance les répétitions avec les boucles d'une manière chronologique. À la fin, chaque répétition détectée automatiquement est associée à une boucle spécifique.

Pour l'évaluation d'un taux d'erreur de la détection des répétitions, nous associons chaque segment automatique à un segment de la vérité terrain chronologiquement.

Étant donnée une répétition **i** effectuée dans un segment $\mathbf{RM_i}$ selon la vérité terrain, et $\mathbf{RA_i}$ est le segment correspondant sélectionné automatiquement.

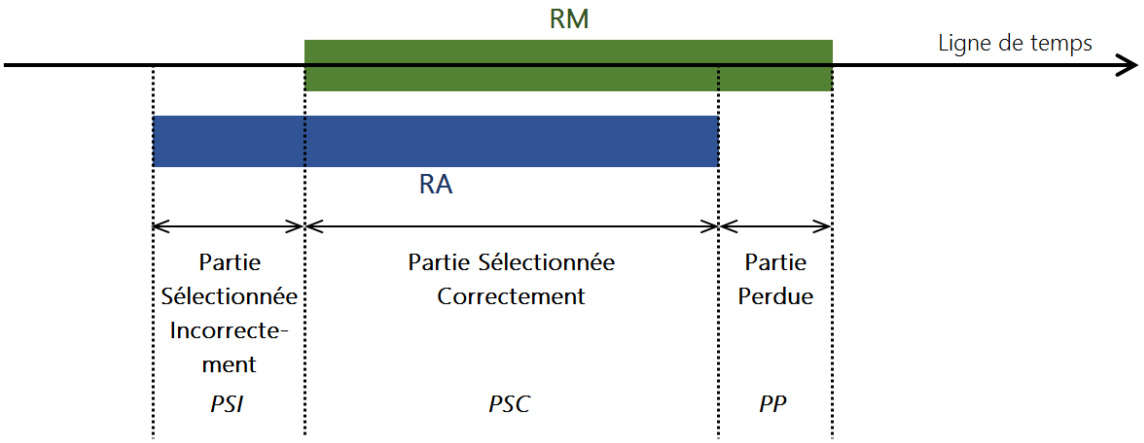


Figure 67 : Calcule de la différence entre chaque RA et et son RM conduit à déduire le taux d'erreur global de méthode.

On calcule les taux d'erreur suivants :

$$T1 = \frac{Partie\ Perdue}{Segment\ correct} = \frac{PP}{RM}$$

Équation 17 : Taux d'erreur 1.

$$T2 = \frac{Partie\ Selectionnée\ Incorrectement}{Segment\ Automatique} = \frac{PSI}{RA}$$

Équation 18 : Taux d'erreur 2.

On déduit le taux d'erreur produit à la répétition **i** :

$$T_i = \frac{(Partie\ Selectionnée\ Incorrectement) + (Partie\ Perdue)}{Union(Segment\ Correct, Segment\ Automatique)} = \frac{PSI_i + PP_i}{union(RM_i, RA_i)}$$

Équation 19 : Taux d'erreur.

Le **Tableau 18** présente les taux d'erreur calculés pour chaque répétition dans notre base de données :

Vidéo	Durée de vidéo (secondes)		Début	Fin	Durée d'occurrence (trame)	Nombre d'occurrences	Durée de la répétition (trames)	Taux d'erreur 1 - T1	Taux d'erreur 2 - T2	Taux d'erreur - T
mhd (1)	116	GT	470	811	54	6	341	0.59	0.07	0.61
		Auto	459	609	50	3	150			
		GT	1000	1990	198	5	990	0.27	0.19	0.38
		Auto	826	1724	179.6	5	898			
		GT	2080	2615	77	7	535	0.00	0.16	0.16
		Auto	1999	2637	79.75	8	638			
mhd (4)	92	GT	575	1300	145	5	725	0.00	0.26	0.26
		Auto	463	1441	140	7	978			
		GT	1440	1745	76	4	305	0.00	0.22	0.22
		Auto	1381	1772	65.1	6	391			
mhd (6)	111	GT	325	1155	207	4	830	0.41	0.07	0.44
		Auto	287	815	176	3	528			
		GT	1478	1875	57	7	397	0.46	0.11	0.50
		Auto	1451	1691	60	4	240			
		GT	2100	2730	157	4	630	0.73	0.00	0.73
		Auto	2148	2316	56	3	168			
roof (1)	74	GT	330	610	47	6	280	0.00	0.07	0.07
		Auto	313	614	25	12	301			
		GT	800	1480	170	4	680	0.42	0.24	0.51
		Auto	676	1194	129	4	518			

		GT	1525	1850	81	4	325	0.00	0.03	0.03
		Auto	1524	1859	83.75	4	335			
roof (3)	48	GT	150	1220	214	5	1070	0.66	0.10	0.67
		Auto	108	515	135	3	407			
sport (8)	32	GT	160	440	35	8	280	0.00	0.04	0.04
		Auto	157	448	32	9	291			
		GT	530	730	50	4	200	0.00	0.10	0.10
		Auto	510	733	27.8	8	223			
sport (1)	44	GT	185	374	31	6	189	0.00	0.14	0.14
		Auto	171	392	18.5	12	221			
		GT	604	878	55	5	274	0.03	0.02	0.05
		Auto	598	871	24.8	11	273			
sport (6)	42	GT	586	790	29	7	204	0.00	0.10	0.10
		Auto	567	794	25.2	9	227			
sport (7)	83	GT	360	636	40	8	276	0.19	0.00	0.19
		Auto	387	610	31	7	223			
		GT	1036	1613	58	10	577	0.07	0.00	0.07
		Auto	1074	1608	53.4	10	534			
		GT	1952	2170	44	5	218	0.01	0.00	0.01
		Auto	1954	2171	36	6	217			
sport (4)	25	GT	142	350	35	6	208	0.00	0.55	0.55
		Auto	114	580	116	4	466			
sport (2)	28	GT	200	540	85	4	340	0.09	0.22	0.28
		Auto	113	509	79.5	5	396			
sport (3)	27	GT	270	570	60	5	300	0.02	0.09	0.11
		Auto	240	564	53.8	6	324			
cuisine (4)	93	GT	635	1090	45.5	10	455	0.04	0.00	0.04
		Auto	635	1074	44	10	439			
		GT	1628	1866	238	15	238	0.28	0.03	0.30
		Auto	1623	1799	16	11	176			
		GT	1885	2210	108	3	325	0.27	0.16	0.36
		Auto	1840	2121	93.6	3	281			

cuisine (1)	109	GT	1075	1720	129	5	645	0.27	0.06	0.30
		Auto	1247	1749	125.5	4	502			
14	**924**						**11837**	**0.18**	**0.11**	**0.27**

Tableau 18: Les taux d'erreur des répétitions dans notre corpus déjà décrit dans la section 4.8.1.2.2 (27 répétitions dans 14 vidéos), où les répétitions appartiennent aux deux types de répétition (voir la section "Vidéos réelles"). On désigne par "GT" (respectivement auto) les informations de la vérité terraine – référence (respectivement limites détectées automatiquement).

On remarque dans le tableau ci-dessus que les taux d'erreur varient entre 0 et 0.73, mais comme on verra dans les histogrammes suivants, les pluparts des taux d'erreur sont acceptables (petites valeurs). D'autre part, il est important de mentionner que les taux d'erreur les plus élevés correspondent aux répétitions du deuxième type qui est plus complexe et attaquant (on a discuté ce point dans la section "Vidéos réelles")

Le Tableau 19 et la Figure 68 présentent les histogrammes des trois types de taux d'erreurs (T, T1 et T2):

Valeur de taux d'erreur	**Nombre d'apparition**			**Pourcentage**		
	Dans "T"	**Dans "T1"**	**Dans "T2"**	**Dans "T"**	**Dans "T1"**	**Dans "T2"**
<0.1	7	16	14	25.93 %	59.26 %	51.85 %
<0.2	13	17	22	48.15 %	62.96 %	81.48 %
<0.3	16	21	26	59.26 %	77.78 %	96.3 %
<0.4	20	21	26	74.07 %	77.78 %	96.3 %
<0.5	21	24	26	77.78 %	88.89 %	96.3 %
<0.6	24	25	27	88.89 %	92.59 %	100 %
<0.7	26	26	27	96.3 %	96.23 %	100 %
<0.8	27	27	27	100 %	100 %	100 %
<0.9	27	27	27	100 %	100 %	100 %
<=1	27	27	27	100 %	100 %	100 %

Tableau 19 : Histogramme des taux d'erreur de chaque type (T, T1 et T2). Par exemple, 96.3% des taux d'erreur T2 sont inférieurs ou égaux à 0.3)

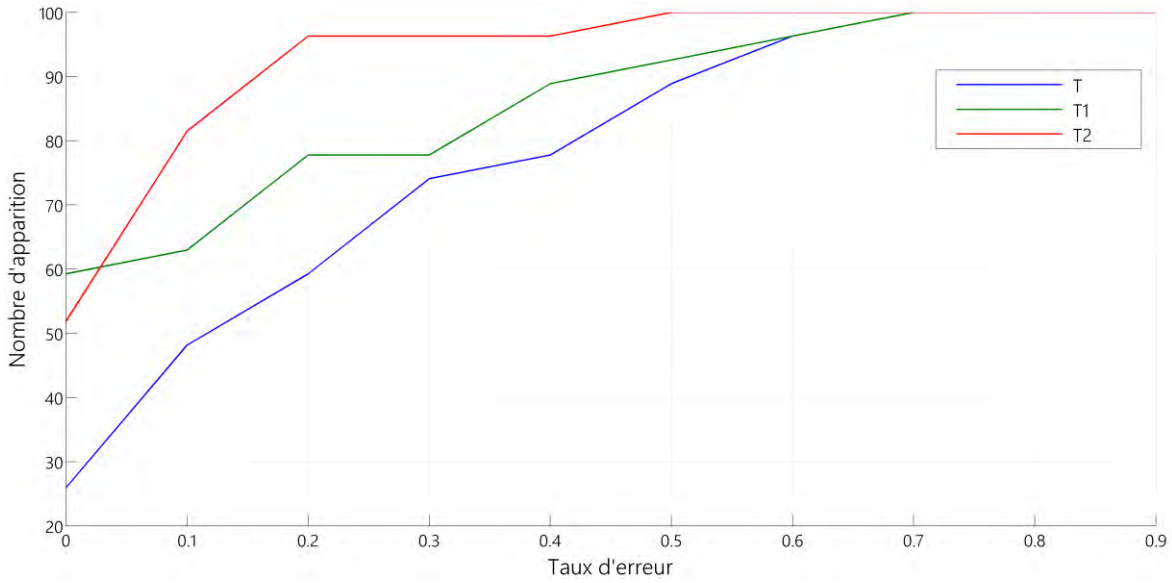


Figure 68: Les sommes cumulatives des histogrammes des taux d'erreur du Tableau 18.

4.8.3.2 Deuxième étape

Nous comparons les taux d'erreur de notre système avec un "Oracle" et un système de base, où :

1) *Un Oracle correspond aux limites des segments sélectionnés par un opérateur humain (autre que l'annotateur de référence) à qui il est demandé d'annoter les segments qui contiennent une répétition. L'Oracle représente le meilleur système capable de localiser les répétitions dans une vidéo. La différence entre les limites de ce système et notre référence montre l'impossibilité de définir une annotation unique et fiable comme étant les limites correctes des répétitions. D'autre part, le but de cette comparaison est de positionner notre méthode par rapport au meilleur système possible.*

2) *Un système de base est ici système dont les limites des segments sont proposées de manière aléatoire sans aucune contrainte. Ce système est supposé représenter un mauvais système de détection. Les limites des segments sont produites de manière aléatoire dans une quantité égale à celles des répétitions réelles dans chaque vidéo. La taille des segments produits est égale à la taille moyenne des segments de la vérité terrain, et nous veillons à ce qu'il n'y ait pas d'intersection entre les segments. A l'inverse de l'oracle, le but est de positionner notre système par rapport à un système trivial et non-exploitable (en termes de résultats).*

Afin de comparer notre système avec les systèmes ci-dessus, nous calculons les taux d'erreur de tous les systèmes selon les équations (Équation 17, Équation 18 et Équation 19), et puis comparer les histogrammes produits par chacun.

Taux d'erreur	Système Oracle	Notre système	Système de base
<0.1	70.4 %	25.9 %	0 %
<0.2	88.9 %	48.1 %	0 %
<0.3	100 %	66.7 %	0 %
<0.4	100 %	74.1 %	3.7 %
<0.5	100 %	81.5 %	11.1 %
<0.6	100 %	88.9 %	14.8 %
<0.7	100 %	96.3 %	14.8 %
<0.8	100 %	100 %	14.8 %
<0.9	100 %	100 %	22.2 %
<=1	100 %	100 %	100 %

Tableau 20: Comparaison des histogrammes des taux d'erreur T de chaque système.

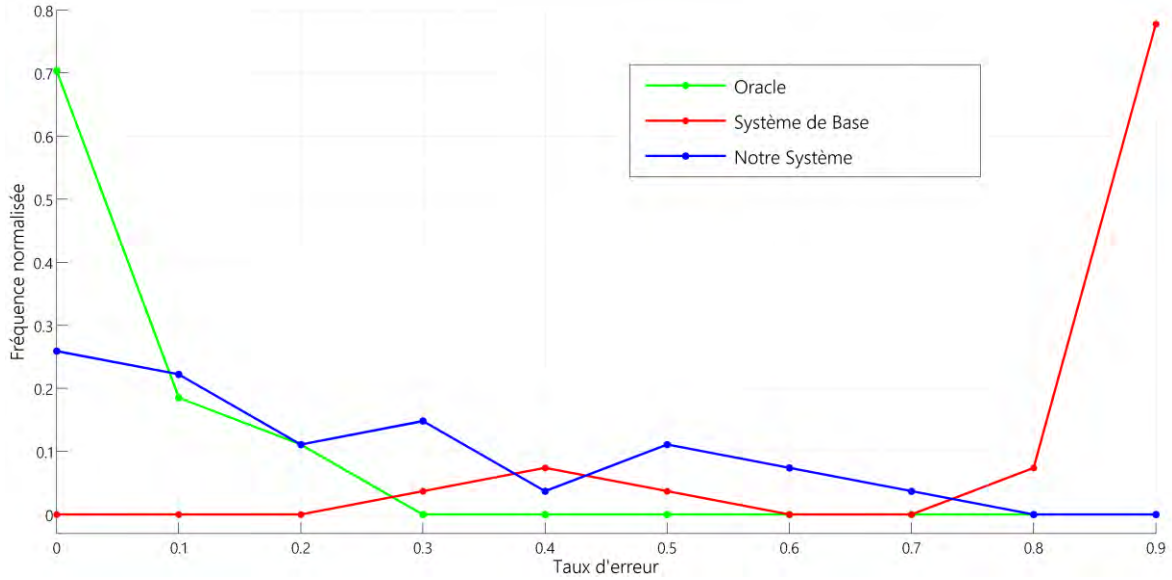


Figure 69 : La fréquence normalisée des taux d'erreurs de chaque système.

4.8.3.3 Fiabilité

Nous examinons maintenant la capacité de notre outil à associer – dans **ANGLEDROIT** - un vote élevé pour l'angle droit qui correspond à un segment fiable. Nous étudions le taux d'erreur de positionnement de chaque segment automatique par rapport au vote associé. Dans cette étude, nous distinguons l'erreur de positionnement des débuts de celle des fins.

Étant donné les segments automatiques et le vote associé à chacun, ainsi que les segments de la vérité terrain (annotés manuellement), tout en considérant que nous extrayons un nombre de segments qui est égal à ceux de la vérité terrain (*cf.* 4.7.3),

nous calculons pour chaque segment automatique, la distance entre son début (resp. sa fin) et le début (resp. la fin) du segment de la vérité terrain. Ensuite, nous associons cette distance au vote calculé pour ce segment. À la fin, nous obtenons deux ensembles des couples (distance, vote) : un pour les débuts et l'autre pour les fins.

Pour chaque ensemble, nous :

→ Trions les votes d'une manière croissante,

→ Divisons l'ensemble en 5 groupes[4] de même taille,

→ Représentons chaque groupe des votes par la moyenne de ses votes et la moyenne des distances correspondantes,

→ Normalisons les votes entre 0 et 1,

Les résultats sont présentés dans la figure suivante :

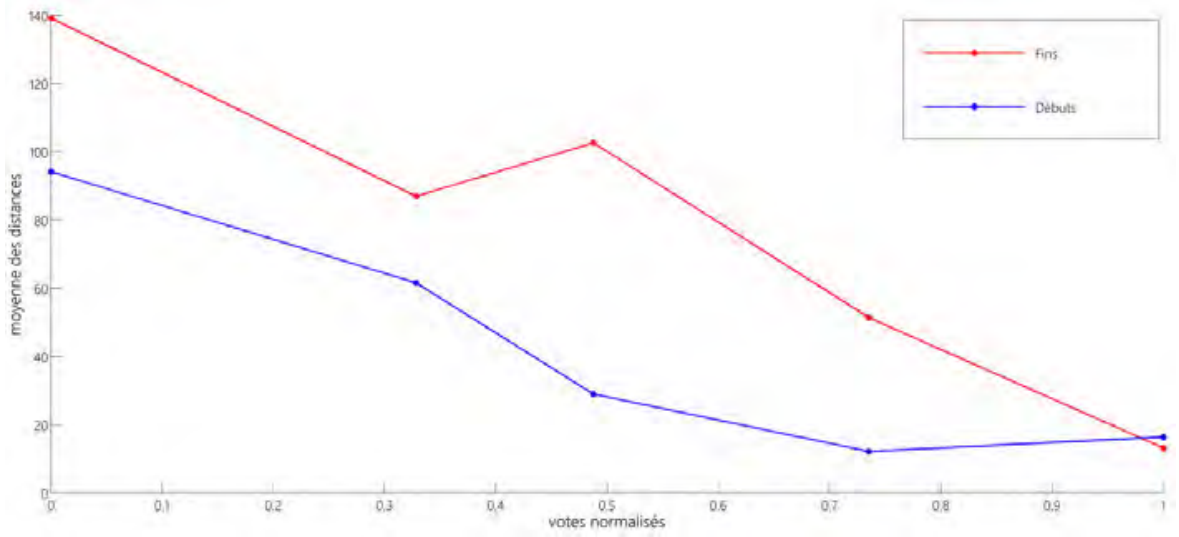


Figure 70 : Moyenne des distances à la limite réelle associée à chaque limite automatique en fonction du vote moyen de l'angle droit normalisé. Le petit nombre de données utilisé pour produire ces courbes (Tableau 13) explique les variations importantes qu'on peut observer. (Taille totale des vidéos : 27720 trames, nombre des répétitions : 27, taille totale des répétitions : 11837 trames, taille moyenne des répétitions : 438.4 trames)

Nous remarquons que les votes les plus élevés (à droite) sont associés aux petites distances. Ce qui signifie que l'erreur de positionnement des segments automatique est plus petite en comparant avec les segments de la vérité terrain. Cette information nous permet de considérer le vote dans la matrice **ANGLEDROIT** comme une mesure de fiabilité. Donc, un segment dont l'angle droit est associé d'un vote élevé/bas induit de petites/grandes distances entre limites réelles et limites automatiques

[4] Le corpus étant petit, nous avons fait le choix de diviser les résultats en seulement 5 groupes.

4.8.3.4 Étude sur le nombre de triangles extraits

Dans cette section, nous étudions les taux des répétitions correctement détectées par rapport au nombre total de triangles extraits. Le but de cette étude est de montrer que l'extraction d'un nombre des triangles qui est plus grand que le nombre de répétitions existant vraiment dans la vidéo, a un effet négatif sur ce taux. Cela doit nous conforter dans notre choix qui consiste à extraire un nombre de triangles égal au nombre de boucles dans le texte. Une étude future peut consister à calculer le nombre de triangles à extraire en fonction du nombre des boucles dans le texte.

Soit **L** le nombre de triangles que l'on souhaite extraire d'une vidéo **v** qui contient **N** répétitions, **T** = $\{t_1, t_2, \ldots, t_L\}$ l'ensemble des triangles extraits, et c_L (resp. e_L) le nombre de triangles dans **T** qui correspondent vraiment[5] (resp. ne correspondent pas) à une répétition dans **v**:

$$C_v = \{ c_L \ ; \text{où } L = N - 3 \rightarrow N + 3\}$$
$$E_v = \{ e_L \ ; \text{où } L = N - 3 \rightarrow N + 3\}$$

Soit les vecteurs suivants (moyenne des taux):

$$\text{Corrects}(t) = \underset{v}{\text{moyenne}}\, C_v(t)$$
$$\text{Erronés}(t) = \underset{v}{\text{moyenne}}\, E_v(t)$$

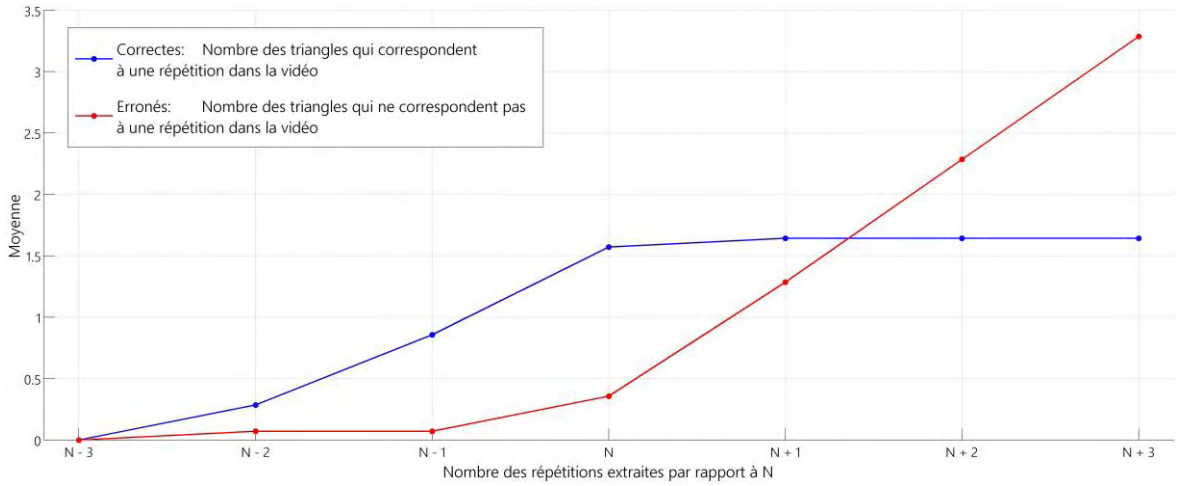


Figure 71 : Moyenne des nombres des triangles qui correspondent (resp. ne correspondent pas) à des répétitions dans les vidéos en fonction du nombre de triangles extraits.

Nous observons bien que l'augmentation du nombre de triangles n'améliore pas les résultats, mais au contraire nous obtiendrons plus des triangles erronés. Cela

[5] L'identification des triangles qui correspondent à une répétition dans la vidéo et ceux ne correspondent pas, est totalement subjective. Dans cette expérimentation, nous effectuons une identification manuelle.

confirme que le choix qui consiste à extraire seulement un nombre défini de triangles (*cf.* 4.7.3), est un choix raisonnable et argumenté.

4.8.4 Limitations

Les bons résultats de notre système comparé aux autres systèmes sur notre corpus ne signifient pas que le système se comporte parfaitement bien dans toutes les situations. Le système de détection des répétitions est proposé et adapté comme étant une partie du système global de synchronisation de deux contenus, l'un textuel et l'autre vidéo. Pour le moins, les hypothèses déjà définies doivent être vérifiées pour que notre système détecte bien les répétitions (voir la section "Vidéos réelles"). Mais de plus, d'autres paramètres peuvent affecter la performance du système d'une façon ou d'une autre sans que nous puissions les contrôler :

4.8.4.1 Similarité des occurrences

Dans la plupart des cas, un opérateur humain peut difficilement répéter la même action d'une manière complément identique (durée d'occurrence et exécution d'action), spécialement quand les occurrences sont très nombreuse ou quand l'action répétée est longue. Cela signifie que la différence entre les occurrences peut apparaître dans la vidéo, ce qui réduit la capacité à caractériser la répétition dans le signal qui la représente. Durant l'application de notre système, deux scénarios peuvent avoir lieu :

- → Au lieu de détecter le segment qui limite toutes les occurrences de la répétition, le système détecte le segment qui limite les occurrences les plus similaires entre elles.
- → Si les occurrences sont très différentes entre elles, alors le triangle correspondant dans la matrice **MATYIN** sera moins marqué et par suite les détails extraits et la position des répétitions seront moins fiables.

L'exemple suivant (Figure 72) correspond à une vidéo qui contient une répétition, mais les occurrences ne sont pas similaires. Toutefois, il existe deux groupes d'occurrences qui sont similaires entre elles. Cela apparait sous forme de deux triangles différents dans la matrice **MATYIN**.

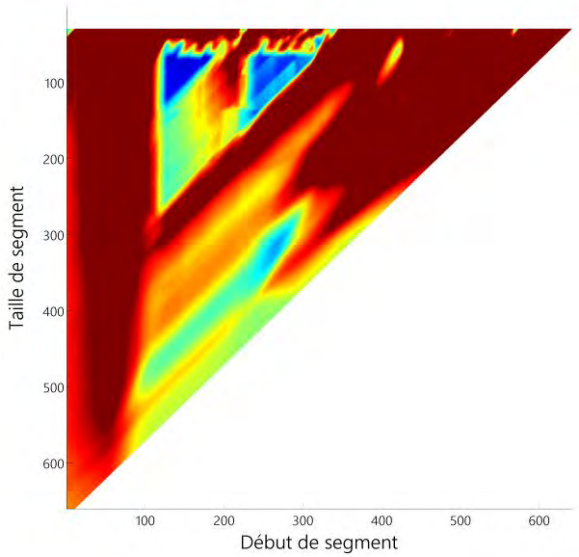


Figure 72: MATYIN d'une vidéo qui contient une répétition d'occurrences non identiques entre elles, mais regroupables en deux groupes.

4.8.4.2 Une partie de la répétition est cachée

Parfois, l'exécution de la répétition ou d'une partie de celle-ci s'effectue d'une manière invisible par la caméra à cause d'un objet qui vient masquer tout ou partie de l'action. Il est impossible dans cas de détecter automatiquement les occurrences cachées de la répétition, ce qui est aussi difficile même d'une manière manuelle. Par conséquence, le système va détecter seulement l'ensemble des occurrences visibles dans la vidéo comme étant la répétition complète.

4.8.5 Conclusion et Intégration dans le système de synchronisation

On a proposé et évalué une technique de détection des répétitions comme une partie du système global visant à synchroniser le contenu d'une vidéo avec le texte qui décrit son contenu. Cette technique extrait automatiquement les limites des segments qui entourent une répétition dans la vidéo. Ces segments sont mis en correspondance avec les boucles dans le contenu textuel.

4.9 Conclusion - Perspectives

Nous avons présenté dans cette section, un outil pour localiser les segments vidéo qui contiennent des actions répétés d'une manière consécutive. Le but de cette méthode est d'associer ces segments aux boucles dans le contenu textuel. Au début, nous avons représenté la vidéo sous la forme d'un signal monodimensionnel qui transcrit le caractère périodique des répétitions.

Le système proposé est capable de localiser les segments à contenus périodiques dans un signal donné. Ce système utilise une technique originalement proposée pour

trouver la fréquence fondamentale d'un signal audio. Nous avons proposé une représentation matricielle dans laquelle les répétitions correspondent à des zones triangulaires de petites coefficients. Le repérage de ces triangles a conduit à localiser les répétitions, ainsi que leurs paramètres (durée et nombre d'occurrences). Plus tard, ce système est testé et évalué sur un corpus dédié.

Nous avons identifié un certain nombre d'améliorations possibles ou de nouvelles pistes d'étude qui pourraient enrichir cet outil :

1) *À l'étape de la représentation de la vidéo sous forme d'un signal, d'autres informations (que la simple intensité) peuvent être utilisées pour prendre en considération des cas plus complexe (caméra mobile, existence de mouvements qui se déroulent ailleurs à l'image que dans la zone ou figure la répétition, etc.). Il peut être judicieux d'utiliser des caractéristiques additionnelles, comme le vecteur et l'orientation des mouvements, l'évolution des gradients ou d'autres caractéristiques encore.*

2) *Dans le cas du traitement de longues vidéos, nous proposons d'appliquer l'outil de calcul des matrices sur des fenêtres glissantes pour contrôler le temps de calcul. La taille des fenêtres peut être déterminée en fonction du type de répétition cherchée. De même, une information explicitant un paramètre d'une répétition recherchée peut réduire le temps de calcul de manière effective. Par exemple, si on peut connaître une estimation de la durée de la répétition, il ne sera pas utile de calculer toute la matrice MATYIN, mais juste une partie d'elle.*

Chapitre 5

SYNCHRONISATION

Comme on l'a mentionné au départ de ce travail, la synchronisation est traitée en deux étapes : d'abord l'extraction des informations de structuration à partir du texte et de la vidéo, et puis la mise en correspondance des informations issues des deux médias. L'extraction s'effectue par des outils (textuels et vidéo) différents et indépendants. Nous ne nous intéressons dans ce travail qu'aux outils de traitement vidéo.

Pour cela, nous avons proposé deux outils qui segmentent les actions unitaires et les répétitions exécutées dans la vidéo. Un outil de détection des segments similaires est aussi présenté en bref dans la section 4.2.4 du Chapitre 4. Selon nos hypothèses de travail formulées dans le Chapitre 2 : "Contexte : État de l'Art, Définitions et Hypothèses", nous ne prétendons pas proposer des outils qui extrairaient des informations à partir du texte, utiles pour la synchronisation. Néanmoins, nous supposons que des outils développés dans le domaine du traitement du langage naturel pourraient permettre de les récupérer automatiquement. Dans notre cas, nous produisons manuellement ces informations pour poursuivre le développement du processus de synchronisation.

Après avoir récupéré ces informations sur les deux médias, la synchronisation consiste alors à mettre en correspondance les éléments de structure similaire issus à la fois du contenu vidéo et du contenu textuel.

5.1 Hypothèses

Dans le Chapitre 2 (*cf.* 2.4.5), nous avons cité les informations que l'on souhaite avoir à partir du contenu vidéo, et nous avons discuté en bref des problèmes rencontrés pour extraire ces informations d'un texte rédigé en langage naturel. Nous avons proposé de représenter le texte sous forme d'un graphe qui précise l'ordre d'exécution des actions, les positions des conditions, les boucles et les instructions similaires. Auxquels s'ajoutent des informations telles que des indicateurs temporels, des valeurs numériques, *etc*... Par la suite, nous appellerons "**chemin textuel**" un parcours (ou une instance de parcours) de ce type de graphe.

On observe que les procédures textuelles contiennent des instructions dont l'exécution dépend d'une condition donnée, ce qui implique l'existence des plusieurs chemins textuels possibles. Par contre, un enregistrement vidéo ne peut correspondre qu'à l'exécution d'un seul chemin textuel, d'où le besoin d'identifier le chemin exécuté dans la vidéo avant de synchroniser son contenu. Jusqu'à maintenant, nous ne sommes pas en mesure d'identifier le chemin suivi dans la vidéo d'une manière automatique.

La reconnaissance automatique du chemin textuel suivi dans une vidéo donnée n'est pas un problème simple. Elle suppose la reconnaissance des actions exécutées ainsi que la compréhension des instructions textuelles. Nous n'aborderons pas ce problème sous cet angle. Nous en réduisons la complexité en :

- → nous référant à un corpus qui contient des vidéos correspondant à tous les chemins textuels possibles et nous supposons que chaque chemin textuel est exécuté dans plusieurs vidéos.
- → considérant qu'une analyse de contenu peut alors être effectuée pour catégoriser et regrouper les vidéos correspondant aux mêmes chemins textuels.
- → effectuant uniquement l'étude des actions exercées dans les vidéos du même groupe (i.e. correspondant à un même chemin textuel),

Par le fait, notre première hypothèse de travail est la suivante :

Hypothèse VI. *Nous considérons seulement le cas d'une vidéo dont le chemin textuel correspondant est déjà connu.*

Le processus de synchronisation consiste à associer chaque information dans un contenu avec l'information correspondante dans l'autre contenu. Pour atteindre une synchronisation idéale, le sens de cette association doit être vérifié dans les deux directions. Dans la pratique, la fiabilité des résultats issus de l'analyse automatique est variable et devrait être prise en compte dans la prise de décision visant à établir ces liens. Or, en l'absence d'outils ad 'hoc sur l'analyse de texte, nous devons définir une

hypothèse restrictive pour pouvoir aborder ce problème. Nous formulerons l'hypothèse forte suivante :

Hypothèse VII. *Nous considérons que les informations provenant de l'analyse du texte sont fiables à 100 %. Nous les utiliserons comme référence dans le processus d'association.*

Nous attendons du processus d'analyse du texte la production d'une représentation de la procédure suivie sous la forme d'un graphe linéarisé du type :

Figure 73: représentation séquentielle des instructions textuelles.

Dans cette figure, un nœud correspond à une instruction unitaire dont nous devons trouver la correspondance de ses limites. On remarque aussi qu'une boucle est représentée par une séquence de nœuds de manière à correspondre plus directement avec une représentation temporelle linéaire du contenu de la vidéo.

Les principales limites que nous ne traiterons pas concernent :

→ la possibilité qu'il existe des différences entre la procédure textuelle de nature prescriptive et la réalisation effective de cette procédure

→ l'utilisation d'information sur la nature des actions exécutées dans l'enregistrement vidéo.

Concernant le premier point, il pourrait y avoir des cas où l'opérateur humain pourrait choisir de modifier l'ordre d'exécution des actions ou tout simplement en omettre certaines. Concernant le second point, la mise en correspondance pourrait être robustifiée par la connaissance de la nature des actions réalisées. Comme nous n'avons pas cherché à reconnaître les actions et que nous ne disposons que des limites temporelles potentielles de celles-ci, nous ne pouvons envisager qu'une mise en correspondance dans l'ordre strict de l'exécution des actions. De ce fait, nous devons formuler une nouvelle hypothèse restrictive :

Hypothèse VIII. *Toutes les instructions textuelles sont exécutées dans la vidéo de manière séquentielle dans un strict respect de leur ordre chronologique dans la procédure.*

Sur la base de ces hypothèses, le problème de synchronisation est réduit à celui de l'association des nœuds aux segments vidéo détectés.

5.2 Matrices de confiance

D'une manière simple, si on est capable de représenter les informations extraites de la vidéo de manière fiable sous la forme d'une séquence d'actions (graphe), alors une association nœud à nœud est une solution immédiate, mais ce n'est pas le cas. Tout d'abord, l'outil de segmentation que nous avons proposé est conçu pour produire des limites d'actions mais pas directement des segments. De plus, le nombre des limites proposées est plus grand que celui de la vérité terrain. Toutefois, nous disposons d'une information sur la fiabilité des limites dont nous allons pouvoir tenir compte dans le processus de mise en correspondance.

Définition : *Un outil de "type 1" est un outil qui propose des points temporels au lieu de segments vidéo, où chaque point peut correspondre à un début ou une fin d'une action.*

D'autre part, l'outil de détection des répétitions fournit les positions des débuts et des fins des répétitions. Malgré la bonne fiabilité de cet outil (selon le système d'évaluation des résultats), les limites extraites ne coïncident pas toujours avec les positions exactes des limites réelles. Par suite, ces limites ne peuvent pas être associées – telles quelles – aux instructions textuelles qui correspondent à une répétition.

Définition : *Un outil de "type 2" est un outil qui propose des segments vidéo qui délimitent l'exécution(s) d'une ou plusieurs instruction(s), en en précisant le début et la fin.*

Nous considèrerons que la notion de fiabilité associée à ces outils revêt deux formes :

- → nous nous intéresserons d'une part à confiance qu'on peut accorder à une limite proposée automatiquement par rapport aux autres,
- → Et d'autre part à la confiance que nous pouvons avoir globalement dans l'outil mis en œuvre.

Partant de là, nous proposons de traiter la fusion des informations provenant de plusieurs types d'outil en prenant en considération de manière différente ces facteurs de confiance.

Pour cela, nous proposons une méthode pour synchroniser les limites et les segments vidéo avec les instructions textuelles correspondantes à l'aide d'une matrice que nous avons baptisée : "**Matrice de confiance**" (**MC**). Dans cette méthode, chaque

outil est associé à sa propre matrice de confiance. Les matrices issues des différents outils sont ensuite fusionnées avant d'en extraire les associations cherchées.

5.2.1 Plan de synchronisation

Le schéma de la Figure 74 résume les informations du paragraphe précédent.

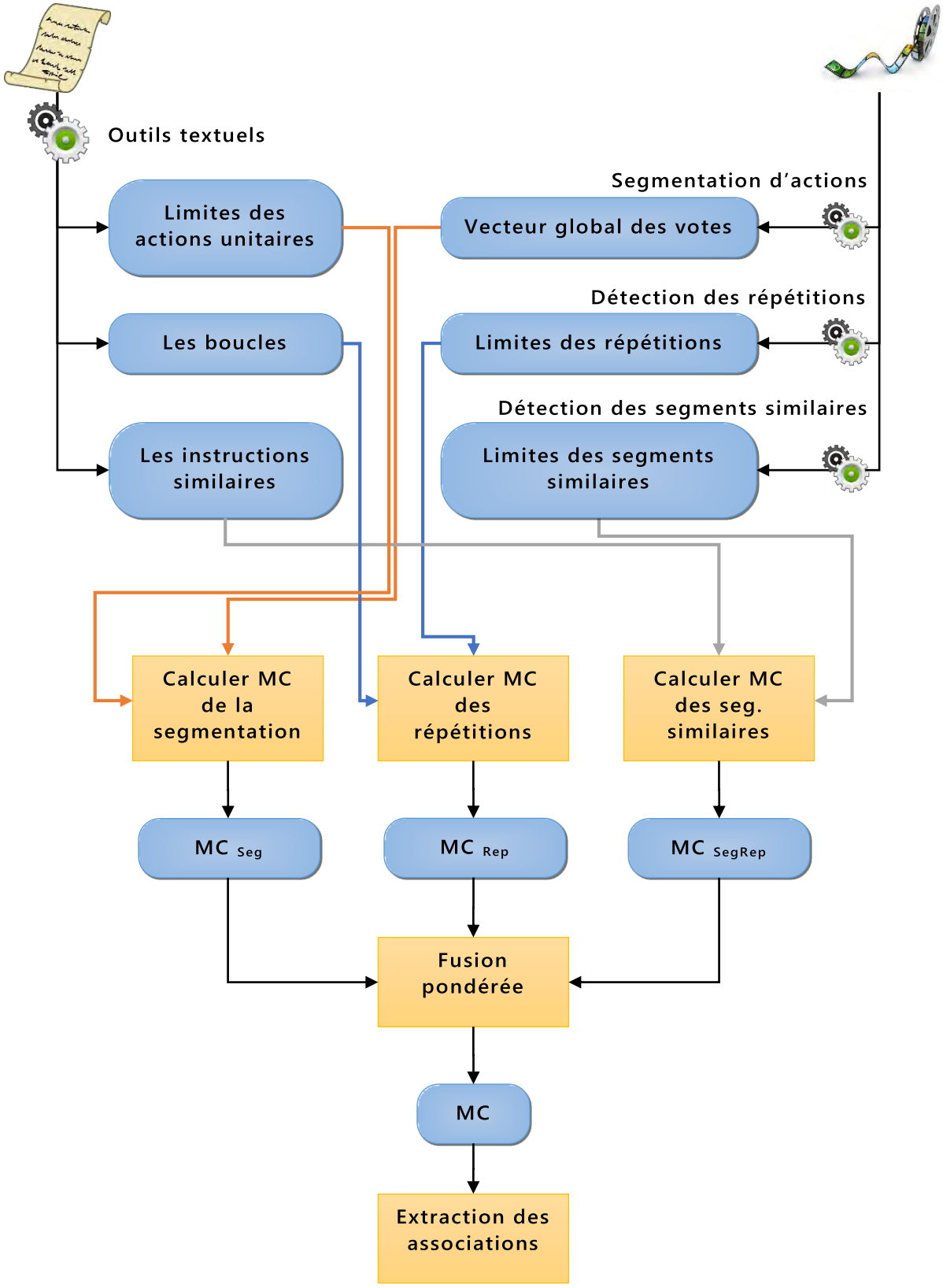


Figure 74 : plan de synchronisation.

5.2.2 Définition

Une "**Matrice de Confiance**" (**MC**) rend compte des correspondances possibles entre les résultats fournis par un outil d'analyse de la vidéo et les instructions du texte. Nous proposons le formalisme suivant pour la construction de cette matrice :

1) ***L'axe vertical représente la ligne de temps*** *sur laquelle seront placés les points temporels issus de l'analyse de la vidéo.*

2) ***Les colonnes représentent les limites textuelles*** *: la séquence des nœuds du graphe représentant les instructions textuelles est traitée dans cette matrice comme une liste ordonnée de numéro des limites (la colonne "1" est le début du premier nœud, la colonne "2" est sa fin, la colonne "3" est le début du deuxième nœud, et ainsi de suite). Nous supposons ici que chaque instruction correspond à une action d'une durée minimale d'exécution (pour le moins non-instantanée) de sorte que chaque instruction corresponde à un début et une fin d'exécution distincts.*

3) ***$MC_{v,t}$ est un coefficient de confiance*** *: cette valeur reflète le niveau de confiance qu'on peut avoir dans la mise en correspondance du point temporel* ***v*** *avec la limite textuelle* ***t***.

$$\begin{array}{cc} & \boxed{\text{Texte}} \\ \boxed{\text{Vidéo}} & \begin{matrix} v_1 \\ \vdots \\ v_N \end{matrix} \end{array} \begin{pmatrix} t_1 & \cdots & t_n \\ MC_{1,1} & \cdots & MC_{1,n} \\ \vdots & \ddots & \vdots \\ MC_{N,1} & \cdots & MC_{N,n} \end{pmatrix}$$

Dans cette matrice, une valeur élevée du coefficient $\mathbf{MC_{v,t}}$ indique que l'association de la limite vidéo **v** à la limite textuelle **t**, est particulièrement plausible. Cette valeur est calculée en fonction de chaque outil en se basant sur ses informations extraites et sa confiance à la position du point **v**.

Pour opérer la synchronisation, nous disposons donc d'autant de matrices de confiances que nous avons d'outils d'analyse de structure de la vidéo. La mise en correspondance est alors établie en portant une attention particulière aux lignes et aux colonnes ayant les coefficients les plus élevés.

Le principe du système que nous avons cherché à mettre en œuvre est qu'il permette d'intégrer la contribution d'autres outils que ce que nous avons présentés précédemment, capables d'extraire des informations complémentaires sur la vidéo, comme par exemple un outil de reconnaissance d'un ensemble des gestes spécifiques. Si cet outil est capable de délimiter les segments vidéo qui contiennent des gestes cherchés, on peut alors imaginer qu'il sera possible d'associer ces segments aux instructions qui impliquent l'exécution de ces gestes. Dès lors, la matrice pourra être prise en compte dans la méthode de fusion.

Dans ce qui suit, nous proposons une méthode pour la construction de la matrice de confiance d'un outil donné, en se basant sur les informations extraites. Puis nous verrons comment cette méthode peut être appliquée à nos outils. Enfin, la méthode de fusion des matrices sera proposée et discutée, ainsi que la technique suivie pour extraire les associations des limites de segment.

5.2.3 Production des coefficients

Dans ce qui suit, nous considérons un outil qui propose des associations entre des limites textuelles et les points temporels correspondants dans la vidéo. Ces associations sont calculées par l'outil en fonction des informations extraites.

*Chaque outil "**O**" fournit des associations vidéo-texte sous forme de couples (numéro de trame, numéro de limite textuelle).*

$$\mathbf{Associations_O} = \{(\mathbf{v},\mathbf{t})\ ;\ \mathbf{v} \in \mathbf{V_O}\ \mathbf{et}\ \mathbf{t} \in \mathbf{T_O}\}\ ;$$

où :

→ Vo est l'ensemble des points temporels détectés dans la vidéo et,

→ To est l'ensemble des limites textuelles à associer par l'outil O.

Équation 20

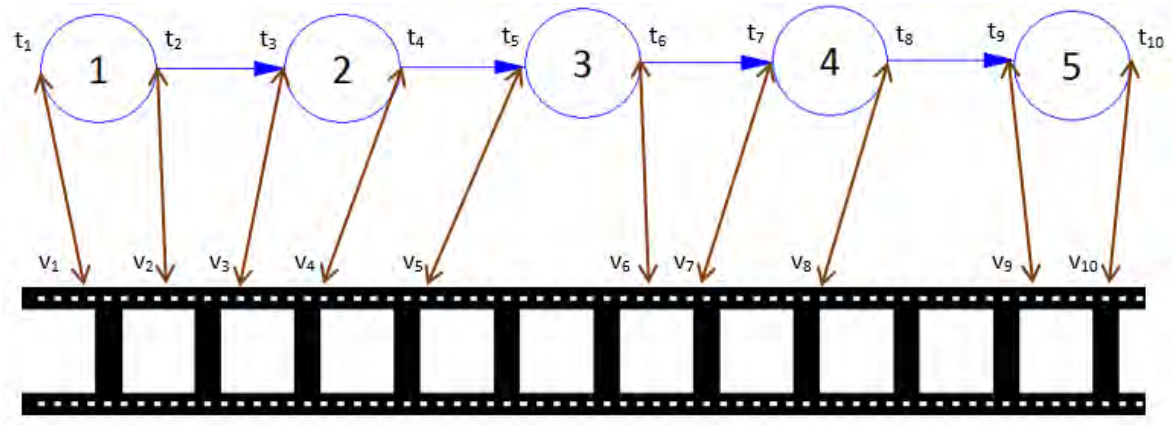


Figure 75 : exemple d'un outil qui propose des associations entre les limites textuelles et les points temporels correspondants. Ci-dessus, le chemin textuel montre les numéros des limites textuelles à associer (t_i). Au-dessous, la ligne de temps montre les points temporels détectés par l'outil (v_i). Une association (t_i, v_i) signifie que - selon cet outil - la limite t_i correspond au point temporel v_i.

Considérons l'Hypothèse VII et supposons dans un premier temps que nous sommes capables d'extraire toutes les limites d'action, chaque limite textuelle doit dans ce cas a une correspondance parmi les limites vidéo. Sous de telles hypothèse, ce fait serait représenté dans la matrice de confiance par un coefficient "1" positionné sur la

ligne et la colonne correspondantes. L'utilisation d'un "1" vient du fait que chaque colonne a certainement une correspondance parmi les limites vidéo car les limites textuelles sont fiables (voir l'Hypothèse VII) et elles forment une référence dans notre système de synchronisation (donc, le "1" correspond à la colonne de l'association). Par conséquent, la somme des coefficients à chaque colonne est toujours égale à "1".

Dans ce qui suit, nous utilisons les limites textuelles comme référence. Nous définissons alors la fonction suivante de remplissage de la matrice de confiance d'un outil donné "**O**" (Où, N est le nombre de trames dans la vidéo) :

$$MC_{v,t} = \begin{cases} 1, & (v,t) \in Associations_O \\ 0, & t \in T_O \text{ et } v \notin V_O \\ 1/_N, & t \notin T_O \end{cases}$$

Équation 21

Selon la fonction de l'Équation 21, l'outil "**O**" remplit chaque entré **(v, t)** par un coefficient qui reflète la possibilité que la limite vidéo "**v**" correspond à la limite textuelle "**t**", dans le cas où $t \in T_O$. Dans le cas où l'outil n'a pas d'informations à propos de la limite textuelle "**t**" ($t \notin T_O$), le coefficient "1" de cette limite (colonne "**t**") est distribué d'une manière équiprobable sur toutes les lignes. Une autre proposition consistant à remplir ces colonnes **(v, t)** par des zéros pose un problème quant à la signification d'un zéro dans la matrice. Dans l'absolu, mettre une valeur zéro dans une cellule **(v, t)** signifie qu'on est sûr que la ligne "**v**" ne correspond pas à la colonne "**t**". Au cours de ce chapitre, nous proposerons une méthode de synchronisation qui fusionne les matrices de confiances de tous les outils selon leurs fiabilités. À cette étape du processus, une valeur qui n'est pas fiable affecte négativement les résultats de synchronisation. La proposition déjà décrite, et sur laquelle nous reviendrons dans le Chapitre 6 : "Conclusion et Perspectives" consiste à distribuer le coefficient "1" de chaque colonne $t \notin T_O$ de manière uniforme sur toutes les lignes.

À ce point, chaque outil disponible a sa propre matrice de confiance qui est construite et traitée d'une manière indépendante des autres outils. À la fin de l'étape de construction, la matrice obtenue par chaque outil est fusionnée avec les autres matrices pour obtenir une matrice finale de confiance qui contient les associations entre les limites textuelles et les limites vidéo.

+	$t1$	$t2$	$t3$	$t4$	$t5$	$t6$
1	0	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v1$	**1**	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	0	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v2$	0	**1**	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	0	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v3$	0	0	**1**	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	0	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v4$	0	0	0	**1**	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	0	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v5$	0	0	0	0	**1**	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	0	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v6$	0	0	0	0	0	**1**
⋮	⋮	⋮	⋮	⋮	⋮	⋮
N	0	0	0	0	0	0

Figure 76 : exemple d'une matrice de confiance qui correspond à un outil qui a des propositions de correspondance entre certaines des limites vidéo et toutes les limites textuelles. La vidéo est formée de N trames et le texte est formé de trois instructions (6 limites).

+	$t1$	$t2$	$t3$	$t4$	$t5$	$t6$
1	0	0	$1/N$	$1/N$	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v1$	**1**	0	$1/N$	$1/N$	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	0	0	$1/N$	$1/N$	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v2$	0	**1**	$1/N$	$1/N$	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	0	0	$1/N$	$1/N$	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v3$	0	0	$1/N$	$1/N$	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	0	0	$1/N$	$1/N$	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v4$	0	0	$1/N$	$1/N$	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	0	0	$1/N$	$1/N$	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v5$	0	0	$1/N$	$1/N$	**1**	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	0	0	$1/N$	$1/N$	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
$v6$	0	0	$1/N$	$1/N$	0	**1**
⋮	⋮	⋮	⋮	⋮	⋮	⋮
N	0	0	$1/N$	$1/N$	0	0

Figure 77 : exemple d'une matrice de confiance, dont l'outil n'est pas capable de trouver les limites vidéo correspondant au début (t3) et à la fin (t4) de l'instruction 2.

5.2.3.1 Remplissage strict

Selon la fonction de l'Équation 10, chaque association **(v, t)** ∈ **Associations**$_O$ proposée par l'outil "**O**" est représentée dans sa propre matrice de confiance, par un coefficient "1" à l'entrée (**v, t**). (Figure 76).

Mais le placement d'un coefficient "1" exclusivement à une seule entrée signifie que l'outil a une confiance absolue dans ses associations. Or, nous savons que malgré la bonne fiabilité des outils vidéo proposés, les résultats ne sont pas complètement fiables. Ceci exige la recherche d'une méthode de remplissage avancée qui prend en considération ce facteur.

5.2.3.2 Remplissage pondéré

Nous allons considérer maintenant que l'outil d'analyse vidéo propose un point temporel correspondant à une limite d'action située dans son voisinage. La marge de ce voisinage dépend de la fiabilité de l'outil.

Nous proposons de distribuer le coefficient de chaque association **(v, t)** ∈ **Associations**$_O$ à l'intérieur de la colonne "**t**", de façon que la portion la plus élevée

soit donnée pour la ligne "**v**", et que les autres lignes reçoivent des valeurs d'autant plus faibles qu'elles sont éloignées de "**v**". Cette distribution doit être telle que la somme des coefficients sur chaque colonne "**t**" soit égale à "1", car l'Hypothèse VII assure qu'une colonne a certainement une correspondance parmi les lignes.

Dans notre cas, nous choisissons d'utiliser une fonction gaussienne normalisée centrée sur la ligne proposée, pour distribuer le coefficient de chaque association **(v, t)**. L'écart-type de la distribution dans la colonne "**t**" est calculée en fonction de la fiabilité globale de l'outil et de la confiance accordée à l'association avec le point "**v**". Si la fiabilité d'un outil est élevée (resp. basse), alors l'écart-type doit être plus petit (resp. grand). Par suite, le coefficient placé à la ligne "**v**" sera plus élevé (resp. bas) (Figure 78/Figure 79) et ceux des lignes voisines seront plus bas (resp. élevés).

Dans la Figure 80, nous présentons un exemple d'une matrice de confiance qui correspond à un outil "**O**", dont **Associations**$_O$ **= {(2,t1), (6,t2), (11,t3), (18,t4), (19,t5), (22,t6)}**. Dans chaque colonne "**t**" de cette matrice, on distribue un coefficient de confiance "1" principalement sur la ligne proposée, et sur son voisinage en utilisant une fonction gaussienne (Équation 22).

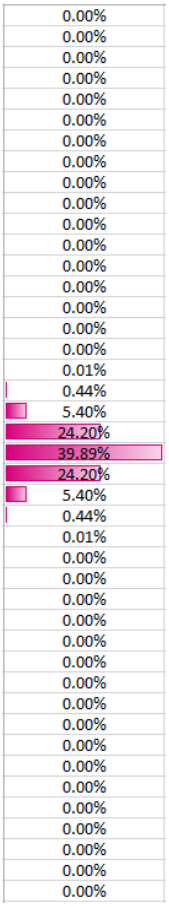


Figure 78 : Exemple de la distribution du coefficient "1" sur les lignes d'une colonne donnée. Cette colonne est dans la matrice d'un outil fiable.

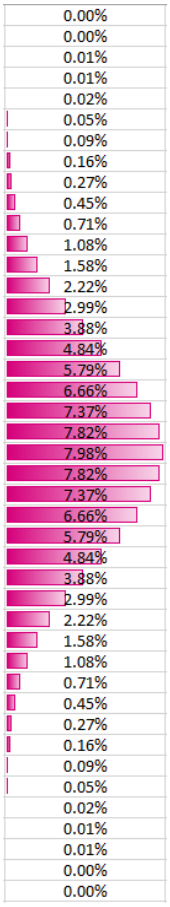


Figure 79: la même association que celle dans la Figure 78, mais celui-là est une colonne dans la matrice d'un outil peu fiable.

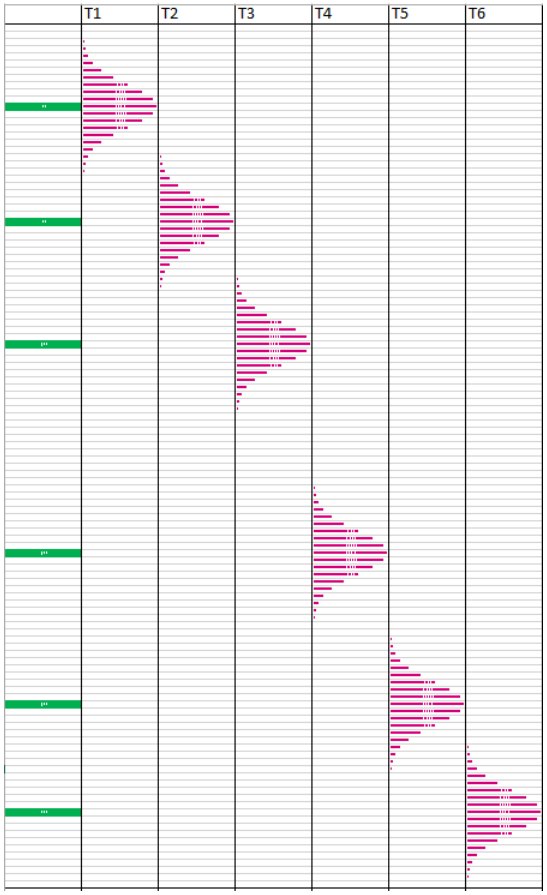


Figure 80: Une représentation graphique d'une matrice de confiance. Les lignes marquées sont les points temporels sélectionnés par l'outil.

Selon l'équation suivante :

$$\int_{-\infty}^{+\infty} \boldsymbol{f(x,\mu,\sigma) = 1}$$

Équation 22

Où, f est la fonction gaussienne normalisée

Dans notre cas, nous visons à distribuer le coefficient "1" d'une colonne **t** sur toutes ses lignes, donc il suffit de calculer la valeur de cette fonction à chaque ligne **v** dans cette colonne (**v** $= 1 \rightarrow$ **N**, où **N** est le nombre des lignes).

$$\mathbf{MC_{v,t}} = \begin{cases} \mathbf{f(v,\ \mu_t, \sigma\ )}, & \mathbf{si\ t \in T_O} \\ \mathbf{1/_N}, & \mathbf{si\ t \notin T_O} \end{cases}$$

Équation 23

Où,

- → **N** est le nombre des trames dans la vidéo,
- → μ_t est le numéro de trame (ligne) associé par l'outil à **t,**
- → $(\mu_t, t) \in \mathbf{Associations_O}$,
- → **σ** est l'écart-type qui contrôle l'étalement de la gaussienne.

L'Équation 22 n'est vraie que dans le cas continu, Nous adaptons le calcul de la fonction gaussienne dans l'intervalle discret [1, **N**]. En particulier, après le remplissage de la matrice selon l'Équation 23, nous normalisons les coefficients de chaque colonne de manière à ce que la somme de chacune soit égale à "1".

5.2.3.2.1 Pourquoi un gaussienne ?

Dans les chapitres précédents, nous avons évalué les outils en calculant les histogrammes des distances entres les points temporels automatiques et les limites des actions réelles (*cf.* Figure 38 et Figure 70). Ces courbes prennent la forme d'une "demi-gaussienne", ce qui signifie que les limites automatiques sont distribuées autour des limites réelles selon une loi normale.

Le petit volume du corpus nous ne permet pas d'affirmer que la distribution suit une loi normale parfaite. Dans le cas d'un corpus plus représentatif, il serait probablement possible de montrer que la distribution des distances ne suit pas une loi gaussienne parfaite, et par suite, l'utilisation d'un gaussienne pour distribuer les coefficients pourrait ne plus être une solution optimale. Dans ce cas, nous proposons de distribuer les coefficients des colonnes en suivant la distribution calculée expérimentalement pour chaque outil.

Actuellement, nous choisissons d'utiliser une distribution gaussienne centrée sur le point automatique, dont l'écart-type est propre à chaque outil.

5.2.3.2.2 Exemple

Soit un contenu textuel qui consiste à effectuer une distribution spécifique de cartes à jouer :

1) *Couper les cartes déposées sur la table en deux paquets,*
2) *Prendre en main une partie des cartes du deuxième paquet,*
3) *déposer 10 cartes devant le premier joueur,*
4) *Prendre en main une partie des cartes du deuxième paquet et déposer-la au-dessus du premier paquet,*

5) Prendre en main les cartes restantes du deuxième paquet,

6) Déposer 10 cartes devant le deuxième joueur

Nous remarquons que le texte ci-dessus contient six instructions, dont deux sont des répétitions en boucle (3 et 6). Notre but consiste alors à trouver dans la vidéo correspondant à l'exécution de cette procédure, les 12 points temporels qui correspondent aux débuts et aux fins de ces instructions.

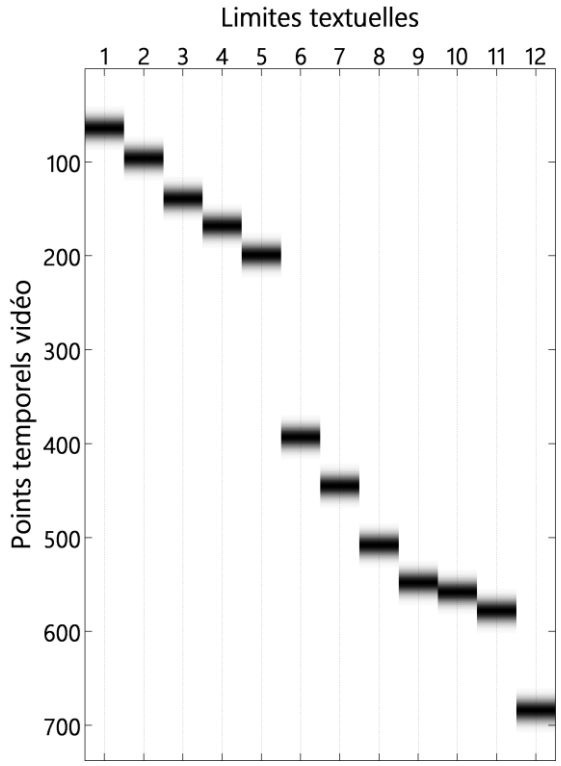


Figure 81 : Matrice de confiance de l'outil d'association correspondant à la segmentation des actions. La couleur noire (resp. blanche) représente des grands (resp. petits) coefficients.

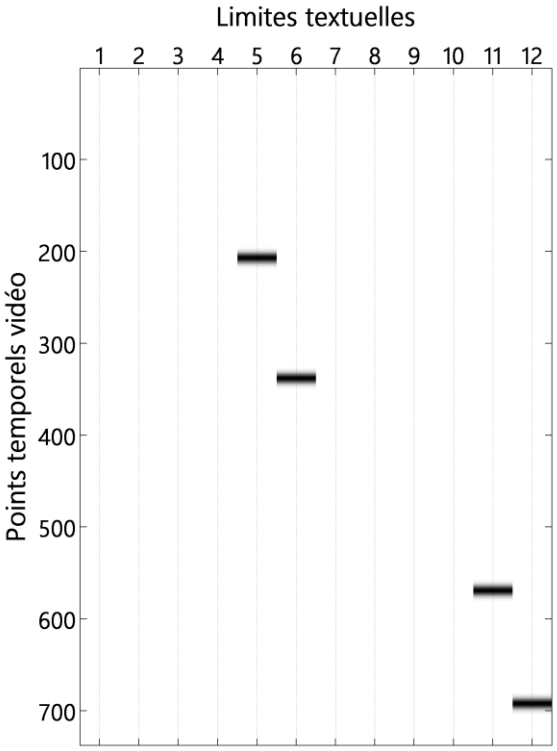


Figure 82 : Matrice de confiance de l'outil de détection des répétitions. On note qu'il est difficile de différencier les zéros des autres très petits coefficients (négligeables par rapport aux coefficients élevés). La couleur noire (resp. blanche) représente des grands (resp. petits) coefficients.

La Figure 81 donne un exemple de matrice de confiance de l'outil d'association correspondant aux segmentations en actions. Nous notons que cet outil propose des associations pour toutes les limites textuelles. La Figure 82 visualise le contenu de la matrice de confiance de l'outil de détection des répétitions en boucle. Comme cet outil ne propose que des associations pour les limites des boucles, nous remarquons que seules les colonnes qui correspondent à ces limites comportent des coefficients élevés. Dans cet exemple, nous supposons que l'outil de détection des répétitions est plus fiable que l'outil de segmentation, pour cela nous utilisons une valeur de σ plus faible pour les répétitions que pour la segmentation.

5.2.3.2.3 Fiabilité générale de l'outil

L'expression de la fiabilité d'une méthode requiert une procédure détaillée qui évalue la méthode dans toutes les situations possibles afin d'en cerner le comportement. Or le volume de notre corpus n'est pas suffisant pour mener cette étude de manière exhaustive.

Nous choisissons toutefois d'utiliser une valeur de **σ** basée sur les expérimentations effectuées sur notre corpus en considérant que cette démarche peut être généralisée sur des corpus plus larges. Nous calculons – pour chaque outil – les distances entre les limites réelles (référence) des actions et les limites automatiques. Nous proposons de fixer la valeur de **σ** en fonction des distances ainsi calculées :

$$\sigma = \text{écart-type (distances)}$$

Équation 24

Dans ce qui suit, nous calculons a priori la valeur **σ** de chaque outil, et puis nous l'utilisons comme une valeur par défaut pour remplir chaque matrice de confiance. Plus tard, nous proposerons de calculer un σ pour distribuer le coefficient "**1**" sur chaque colonne. Sa valeur sera cette fois fonction de la fiabilité estimée de l'association correspondante.

5.2.3.2.4 Fiabilité d'une association

Les associations produites par un outil n'ont pas toujours la même fiabilité. Du point de vue de l'outil, une association peut être considérée comme étant plus fiable qu'une autre. Dans ce cas, il est possible pour chaque colonne, d'adapter la valeur du paramètre **σ** en fonction de la confiance accordée par l'outil de segmentation au point temporel associé. D'où la fonction de remplissage modifiée :

$$MC_{v,t} = \begin{cases} f(v,\ \mu_t, \sigma_t), & \text{si } t \in T_O \\ 1/N, & \text{si } t \notin T_O \end{cases}$$

Équation 25

où,

- → **f** est une fonction gaussienne
- → **N** est le nombre des trames dans la vidéo,
- → μ_t est le numéro de trame (ligne) associé par l'outil à **t,**
- → $(\mu_t, t) \in \textbf{Associations}_O$,
- → σ_t est l'écart-type qui contrôle l'étalement de la gaussienne.

Par exemple, nous avons déjà montré durant l'évaluation de l'outil de segmentation en actions que les points temporels sélectionnés n'ont pas tous la même fiabilité (*cf.* section 3.6.2 - Figure 39). Un point correspondant à un vote élevé est un point plus fiable qu'un autre ayant un vote plus petit. Par suite, nous pouvons accorder davantage de confiance dans l'association du premier point avec une limite issue du texte. Dans les sections suivantes, nous discuterons d'une méthode de calcul

permettant de rendre compte de l'impact de cette confiance sur la valeur du paramètre **σ** pour chaque type d'outil.

Dès lors, chaque outil d'association calcule sa propre matrice selon les informations qui lui sont fournies et ses associations proposées, de manière indépendante avec les autres outils. Nous pouvons imaginer qu'il sera alors possible d'intégrer ultérieurement d'autres résultats d'analyse et de segmentation de la vidéo permettant d'améliorer le processus de synchronisation.

La procédure générale de construction de la matrice d'un outil donné est comme la suivante :

1) *Récupérer les limites/segments produits (si possible en surnombre) par l'outil d'analyse de la vidéo,*
2) *Produire pour chaque limite/segment une information de fiabilité (vote, contraste, etc.),*
3) *Calculer l'écart-type entre les limites ou les bornes de segments réelles et automatiques*
4) *Produire l'association "limite textuelle"/"limite vidéo",*
5) *Répartir le poids sur les trames vidéo voisines de celles qui ont été détectées selon une distribution gaussienne*[6] *ayant pour paramètre l'écart-type précédemment calculé.*

Dans ce qui suit, nous construirons les matrices correspondant aux outils que nous avons proposés précédemment. Nous verrons ensuite comment nous pouvons fusionner toutes les matrices calculées en une seule matrice finale de confiance. Cette méthode prend en considération la fiabilité de chaque outil utilisé.

5.2.4 Matrice de l'outil de segmentation d'actions (outil de type 1)

De manière à pouvoir généraliser les propositions qui suivent, nous distinguerons deux types d'outils que nous appellerons Type 1 et Type 2. La différence entre ces deux types est que les outils du premier type proposent des points temporels sans être capable de distinguer les débuts et les fins d'actions. Les outils du deuxième type proposent des points temporels mais en distinguant les débuts et les fins (débuts et fins des segments). Pour le premier groupe, nous considérons qu'une limite donnée peut correspondre à n'importe quelle limite réelle située à proximité. Pour le deuxième groupe, les points qui représentent des débuts (resp. des fins) des répétitions ne peuvent être associés qu'aux débuts (resp. fins) des boucles dans le texte. Ces

[6] La distribution n'est pas forcement gaussienne pour tous les outils, une étude statistique doit avoir lieu pour déduire la distribution correspondante. Pour nos outils, nous avons montré que le gaussien est la distribution la plus correspondante.

contraintes sont prises en considération durant la proposition des associations entre les points temporels et les limites textuelles. Plus tard, ces associations sont utilisées pour remplir la matrice de confiance.

L'outil de segmentation d'actions (de type 1 donc) n'est pas conçu pour fournir des associations mais des points temporels qui représentent une limite d'action. Nous avons choisi de faire en sorte que le nombre des points sélectionnés soit beaucoup plus grand que le nombre des limites textuelles. Or, seules des associations 1-à-1 peuvent être effectuées. À cette fin, nous proposons de choisir uniquement les points temporels les plus fiables pour les associer dans leur ordre chronologique aux limites textuelles correspondantes.

Cela n'implique pas que les autres points identifiés sont complètement ignorés, Ils seront utilisés plus tard comme des solutions alternatives durant l'étape finale de synchronisation (*cf.* section 5.4).

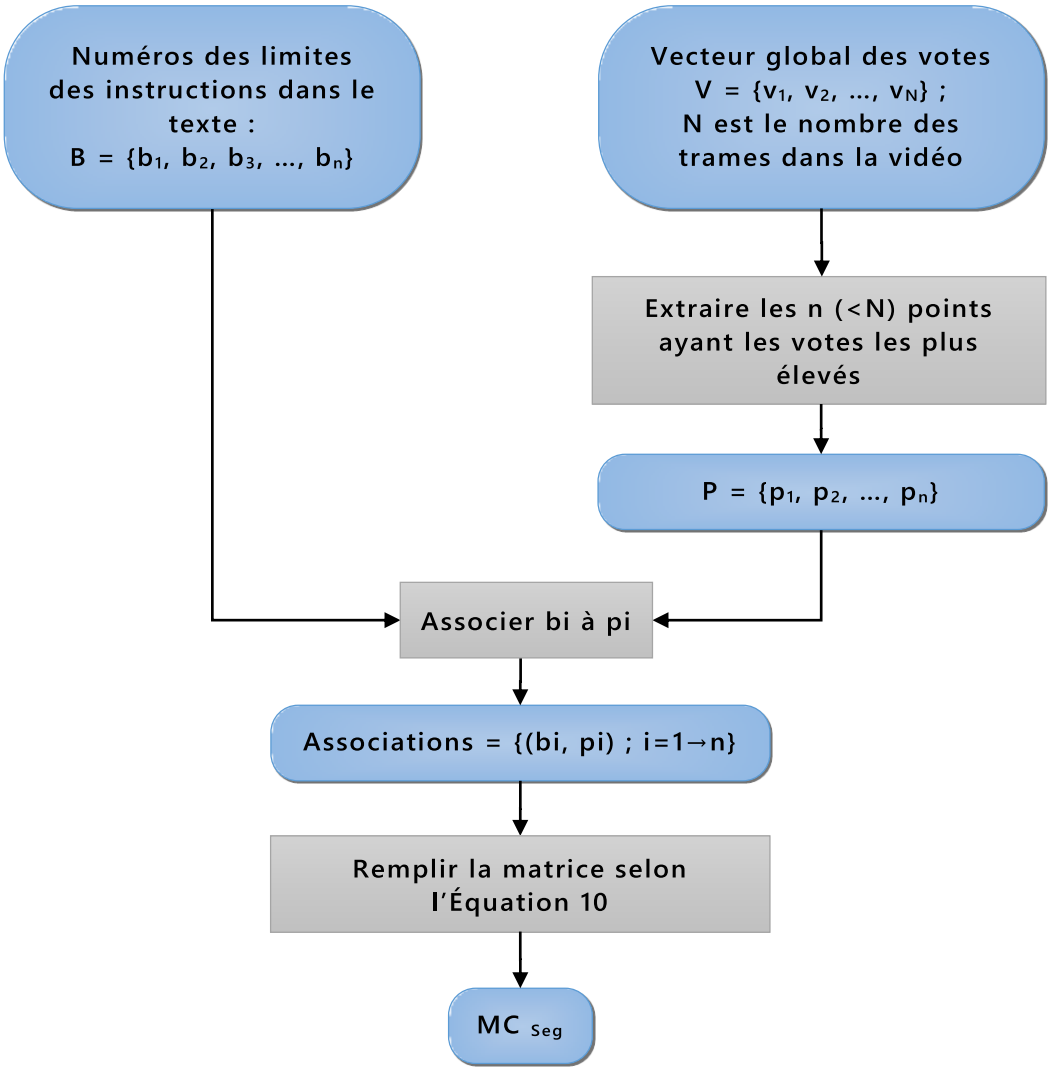


Figure 83 : Remplissage de la matrice de confiance MC de l'outil de segmentation des actions.

$$\mathbf{Assocication_{Seg}} = \{(\mathbf{v_i}, \mathbf{t_i});\ \mathbf{i}: \mathbf{1} \rightarrow \mathbf{n}, \mathbf{v_i} \in \mathbf{V_{Seg}}\ \mathbf{et}\ \mathbf{t_i} \in \mathbf{T_{Seg}}\}$$

Où :

- → **n** est le nombre des limites textuelles,
- → $\mathbf{V_{Seg}}$ contient les **n** points ayant les n votes les plus élevés dans l'ordre chronologique,
- → $\mathbf{T_{Seg}}$ contient les **n** limites textuelles selon leurs ordres d'apparitions dans le texte.

Équation 26

Bien que nous ayons sélectionné les points les plus fiables, cela ne signifie pas pour autant qu'ils ont tous la même fiabilité. Nous proposons alors d'utiliser pour chaque colonne un paramètre calculé en fonction du vote du point temporel correspondant. Le principe général est que, si le point temporel **v** associé à la colonne **t** a un vote relativement élevé, alors la valeur du paramètre **σ** utilisée pour le calcul de la distribution des poids sur la colonne **t** sera relativement petite car le point **v** est considéré comme fiable.

Mais les termes "vote élevé" et "petite valeur du paramètre **σ**" sont des termes subjectifs. Qu'entend-on par petite valeur du paramètre **σ** ? 5, 40, 100 ou 200 trames ? Difficile à dire sauf si on dispose d'une relation entre la valeur d'un vote associé à une limite détectée automatiquement et la distance entre cette limite et la limite réelle de l'action la plus proche. Or, la courbe de la Figure 84 présentée dans le chapitre de la segmentation (*cf.* section 3.6.2 - Figure 39) en actions exprime cette relation.

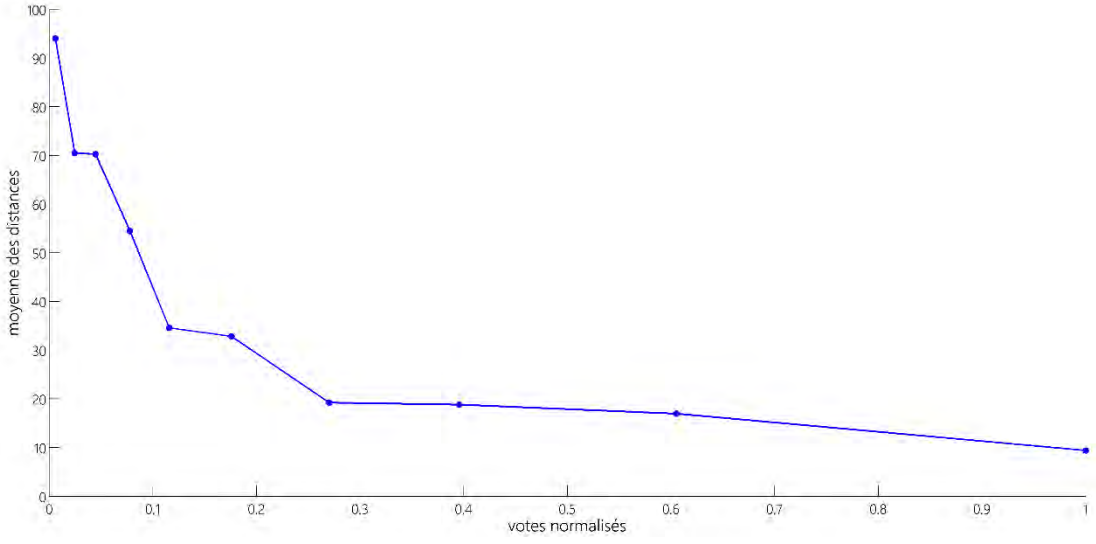


Figure 84: Distribution de l'erreur de positionnement d'une limite d'action en fonction de la valeur (normalisée) du vote associé. La courbe montre que les points ayant des votes élevés sont plus fiables. (Nombre des limites réelles : 218, nombre des limites automatiques : 221, moyenne des distances : 33.0383).

La Figure 84 fournit la moyenne des distances correspondante à chaque vote. Selon l'Équation 24, nous avons besoin de la relation entre l'écart-type et la valeur de vote associé à un point donné. La Figure 85 montre cette relation.

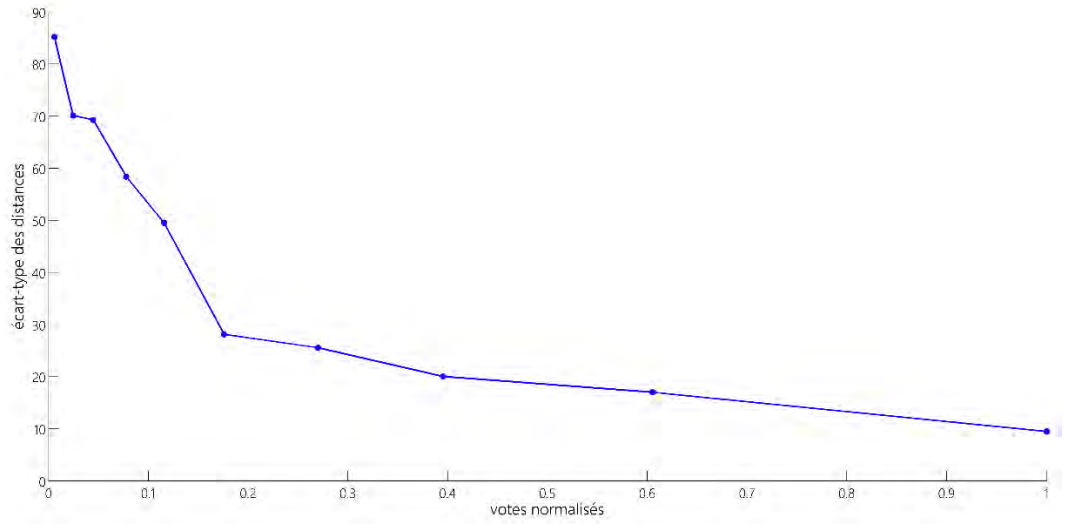


Figure 85: Distribution de l'écart-type des distances en fonction de la valeur normalisée du vote associé. Le même corpus que celui de la Figure 84 est utilisé.

Etant donné une association (**v**, **t**) :

$$\sigma = f1(\alpha),$$

où :

→ **f1** est la fonction qui décrit la courbe de la Figure 85

→ **α** est le vote normalisé associé au point **v** (dans le vecteur des votes)

Équation 27

5.2.5 Matrice de l'outil de détection des répétitions (outil de type 2)

L'outil de détection des répétitions en boucle localise les segments qui contiennent la répétition d'une (des) action(s) en fournissant un début et une fin pour chaque répétition détectée. D'autre part, les informations extraites à partir du contenu textuel identifient les instructions qui consistent en une boucle. Par conséquence, les limites de chaque boucle (le début de la première occurrence et la fin de la dernière occurrence) sont associées aux limites du segment détecté selon l'ordre d'apparition des boucles dans le texte. Selon l'hypothèse d'une analyse textuelle parfaite, nous extrayons de la vidéo autant de segments de répétition que de boucles identifiées dans la procédure. Pour cela, sont sélectionnées prioritairement les répétitions détectées dans la vidéo associée aux plus hauts votes. Cet outil ne fournit aucune information concernant les autres points temporels.

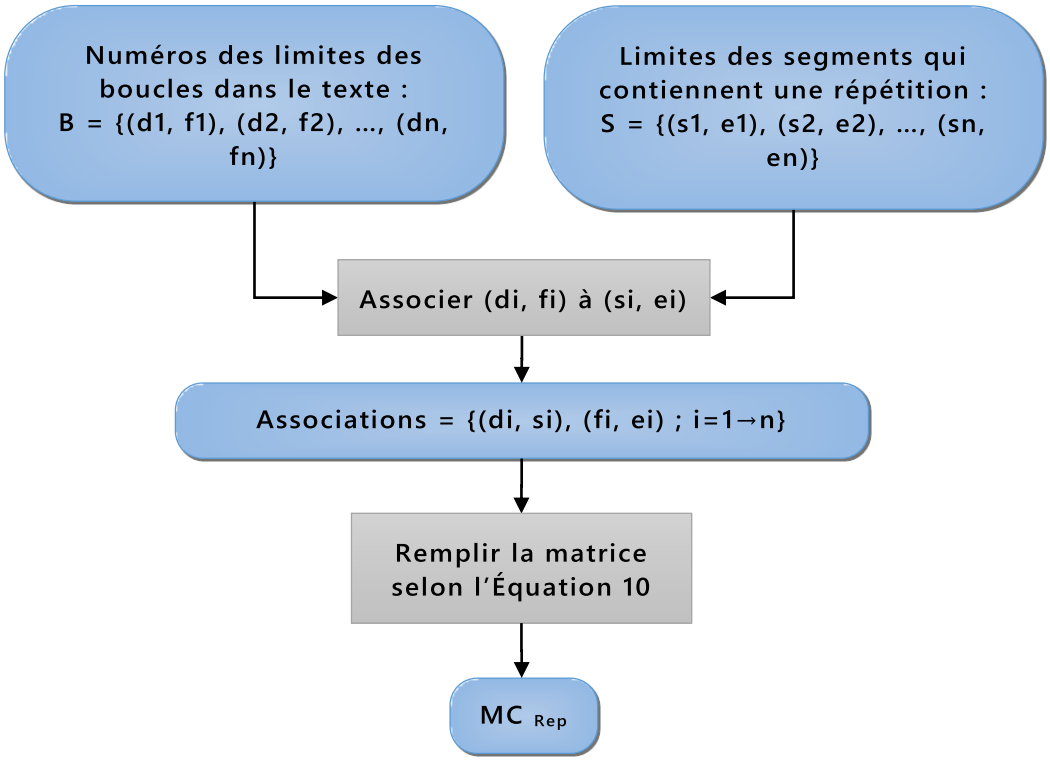


Figure 86 : Remplissage de la matrice de confiance MC de l'outil de détection des répétitions.

Comme on l'a dit, les associations proposées sont représentées dans une matrice de confiance. Cet outil fournit aussi des limites avec une fiabilité variable. Donc, la matrice est remplie aussi selon l'Équation 25, dont la valeur **σ** doit être calculée.

Compte-tenu le fait qu'il s'agit d'un outil de type 2, et à l'instar de la proposition précédente, les distances entre les points détectés automatiquement et la limite réelle la plus proche du même type, nous permettent de calculer la valeur appropriée de **σ.**

La proposition qui suit repose sur des valeurs statistiques nécessitant de nombreuses observations pour qu'elles soient fiables et exploitables. Cette méthode ne pourra cependant pas être mise en œuvre dans la production des résultats présentés plus loin en raison de l'utilisation d'un corpus trop petit pour permettre la production de paramètres représentatifs.

5.2.5.1 Calcul du paramètre σ

Durant l'évaluation de l'outil de détection des répétitions, nous avons calculé une courbe qui montre la relation entre la valeur du vote associé à l'angle droit d'un triangle et la confiance dans le fait que ce segment contienne une répétition (*cf.* section 4.8.3.3). Cependant, ce vote ne reflète pas la confiance dans le fait que le début (resp. la fin) du segment corresponde au début (resp. la fin) de la boucle. Parfois, l'outil ne détecte

qu'une partie de la répétition totale car les occurrences ne sont pas d'une parfaite similarité.

La Figure 87 montre la relation entre l'intensité du vote du point à l'angle droit d'un triangle et l'erreur de positionnement du segment correspondant. Tout en considérant la limite évoquée précédemment sur la validité du vote comme mesure de fiabilité, nous observons qu'un segment dont l'angle droit est associé à un vote élevé/bas induit de petites/grandes distances entre limites réelles et limites automatiques (Figure 87). En tenant compte que le paramètre **σ** est un écart-type, sa valeur est obtenue de la courbe qui montre l'écart-type qui correspond à chaque valeur de vote (Figure 88).

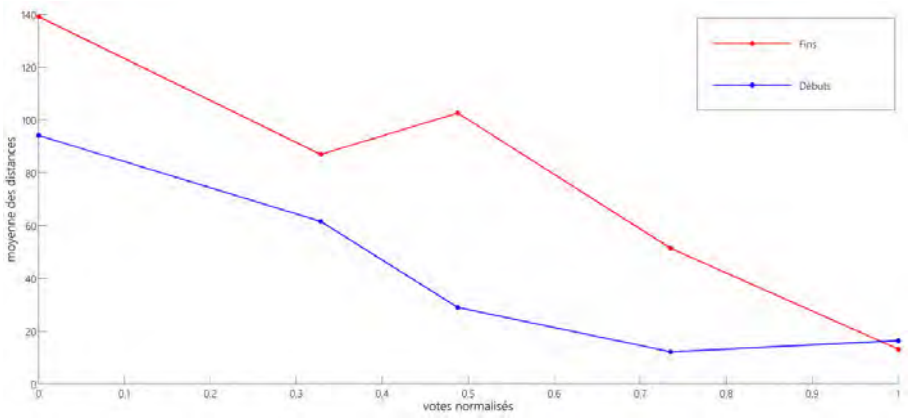


Figure 87 : Moyenne des distances à la limite réelle associée à chaque limite automatique en fonction du vote de l'angle droit normalisé. Le petit nombre de données utilisé pour produire ces courbes explique les variations importantes qu'on peut observer. Taille totale des vidéos : 27720 trames, nombre des répétitions : 27, taille des répétitions : 11837 trames, moyenne des répétitions : 438.4 trames.

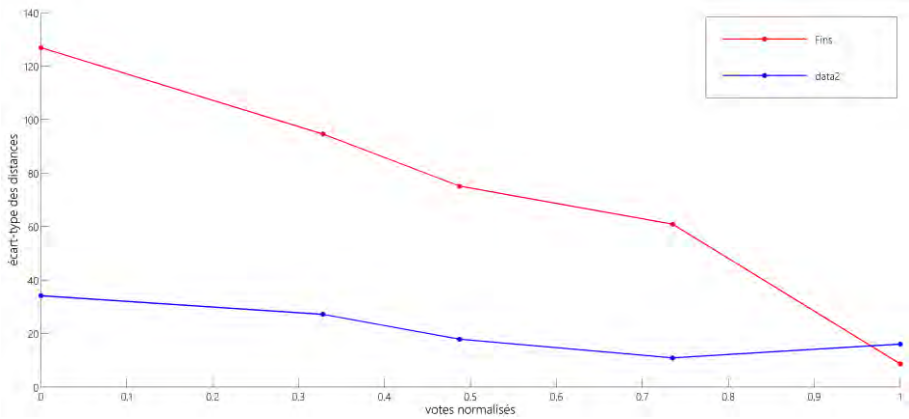


Figure 88 : écart-type des distances à la limite réelle associée à chaque limite automatique en fonction du vote de l'angle droit normalisé. Le même corpus que celui de la Figure 87 est utilisé.

Nous faisons donc le choix d'utiliser les courbes de la Figure 88 pour calculer la valeur de **σ** sur chaque colonne :

$$\sigma_{Débuts} = f2_{Débuts}(\alpha) \text{ et,}$$

$$\sigma_{Fins} = f2_{Fins}(\alpha)$$

où :

→ $f2_{Débuts}$ (resp. $f2_{Fins}$) est la fonction qui décrit la courbe des débuts (resp. fins) dans la Figure 88,

→ α est le vote associé à l'angle droit du segment.

Équation 28

Dans la mesure où ces courbes sont très imprécises en raison de la taille du corpus, nous fixerons dans la partie évaluation la valeur de σ à l'écart-type des distances entre les débuts (resp. fins) des segments détectés et les débuts (resp. fins) réels des segments.

σ débuts (resp. Fins) = écart-type des distances entre les débuts (resp. fins) des segments détectés et les débuts (resp. fins) réels des segments (vérité terrain).

Équation 29

Comme conclusion, la procédure générale du remplissage et distribution des coefficients dans ce type d'outil, est le suivant :

1) *Soit **D** et **F** le début et la fin automatiques d'un segment, et **V** le vote de son angle droit,*

2) *Soit B_d et B_f le début et la fin d'une boucle dans le texte,*

3) *On distribue le coefficient de la colonne B_d (resp. B_f) sur toutes les lignes en utilisant la fonction gaussienne déjà définie, dont :*

   a. *Le centre est la ligne D sur la colonne B_d et sur F sur la colonne B_f,*

   b. $\sigma_{Débuts\ (resp.\ fins)} = f2_{Débuts\ (resp.\ Fins)}(V)$; *(Équation 28)*

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
317	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
318	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
319	0.14%	0.14%	0.01%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
320	0.14%	0.14%	0.44%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
321	0.14%	0.14%	5.40%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
322	0.14%	0.14%	24.20%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
323	0.14%	0.14%	39.89%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
324	0.14%	0.14%	24.20%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
325	0.14%	0.14%	5.40%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
326	0.14%	0.14%	0.44%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
327	0.14%	0.14%	0.01%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
328	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
329	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
796	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
797	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
798	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.01%	0.14%	0.14%
799	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.44%	0.14%	0.14%
800	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	5.40%	0.14%	0.14%
801	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	24.20%	0.14%	0.14%
802	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	39.89%	0.14%	0.14%
803	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	24.20%	0.14%	0.14%
804	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	5.40%	0.14%	0.14%
805	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.44%	0.14%	0.14%
806	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.01%	0.14%	0.14%
807	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
808	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
809	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%
810	0.14%	0.14%	0.00%	0.14%	0.14%	0.14%	0.14%	0.14%	0.14%	0.00%	0.14%	0.14%

Figure 89: la forme générale d'une matrice correspondant à l'outil de détection des répétitions (outil de type 2). Dans cet exemple, Le début et la fin du segment contenant la répétition (D:323 → F:802) sont associés respectivement au début et à la fin de la boucle textuelle (B_d : t3 et B_f : t10). Les coefficients des autres colonnes sont distribués sur toutes les lignes d'une manière uniforme. Dans cet exemple, les deux distributions utilisent la même valeur de σ (Équation 29).

5.2.6 Le cas d'une distribution non-gaussienne

Nous avons déjà dit que les limites automatiques peuvent ne pas être distribuées autour des limites réelles selon une loi normale parfaite, ou même une autre loi différente. Dans ce cas, nous proposons de distribuer les coefficients des colonnes en suivant la distribution déduite.

5.2.7 Matrices des répétitions séparées

Dans la section 4.2.4, nous avons présenté en bref – sans implémentation - une méthode pour localiser les segments répétés à différents moments dans une vidéo (par le moyen d'une matrice de similarité). La présentation de cette méthode n'a pas pour objectif la comparaison, l'évaluation ou l'amélioration, mais elle forme une étude de cas pour imaginer un outil de segmentation des segments similaires, qui nourrit le système de synchronisation et ainsi améliorer sa performance. D'autres méthodes de détection des segments similaires pourraient également être utilisées. Le choix de cette

méthode précisément vient du fait qu'elle permet de repérer les segments similaires d'une manière simple.

Selon les hypothèses formulées sur l'analyse textuelle, nous supposons qu'il existe des outils textuels permettant d'identifier les instructions qui exigent l'exécution d'une action similaire à des instants différents. Nous proposons d'associer chaque ensemble d'instructions similaires à un ensemble de segments vidéo qui délimitent l'exécution de la même action (segments similaires).

Dans le cas où le texte comprend plusieurs ensembles d'instructions similaires, ces ensembles seront associés aux ensembles des segments similaires les plus fiables parmi tous les ensembles sélectionnés, tout en respectant leur ordre chronologique. On pourra exprimer cette fiabilité à l'aide de la mesure de pertinence calculée globalement sur l'ensemble. En pratique :

1) *L'outil textuel fournit une liste des ensembles d'instructions similaires triées selon leurs ordres d'apparition dans le texte,*

2) *L'outil vidéo fournit une liste des ensembles des segments similaires,*

3) *Les ensembles des segments sont triés par ordre décroissant selon leur fiabilité,*

4) *Nous choisissons les n ensembles des segments les plus pertinents, où n est le nombre des ensembles d'instructions similaires dans le texte,*

5) *Les ensembles des segments sélectionnés sont triées selon leur ordre d'apparition dans la vidéo,*

6) *L'ensemble d'instructions i est associé à l'ensemble de segments i (sélectionnés).*

Par exemple, pour un groupe donné de segments vidéo similaires et un groupe d'instructions similaires, les associations s'effectuent de la façon suivante :

→ Le début du premier segment est associé au début de la première instruction,

→ La fin du premier segment est associée à la fin de la première instruction,

→ Le début du deuxième segment est associé au début de la deuxième instruction,

→ Ainsi de suite...

Le remplissage de la matrice de confiance s'effectue selon la méthode présentée dans la section précédente, car cet outil est de type 2 (il fournit les débuts et les fins des segments). Nous devons maintenant distribuer des coefficients de confiance autour des positions qui marquent les limites des segments associés. Dans la mesure où nous n'avons pas pu mener d'expérimentations sur ce sujet, nous supposons seulement qu'il est possible de mesurer un écart-type entre les résultats produits automatiquement et ceux de la vérité terrain. Cet écart-type peut alors être utilisé pour calculer la distribution gaussienne des poids (voir la section précédente).

	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
196	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
202	0.14%	0.14%	0.14%	0.14%	0.88%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
203	0.14%	0.14%	0.14%	0.14%	2.70%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
204	0.14%	0.14%	0.14%	0.14%	6.48%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
205	0.14%	0.14%	0.14%	0.14%	12.10%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
206	0.14%	0.14%	0.14%	0.14%	17.60%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
207	0.14%	0.14%	0.14%	0.14%	19.95%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
208	0.14%	0.14%	0.14%	0.14%	17.60%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
209	0.14%	0.14%	0.14%	0.14%	12.10%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
210	0.14%	0.14%	0.14%	0.14%	6.48%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
211	0.14%	0.14%	0.14%	0.14%	2.70%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
212	0.14%	0.14%	0.14%	0.14%	0.88%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
333	0.14%	0.14%	0.14%	0.14%	0.00%	0.88%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
334	0.14%	0.14%	0.14%	0.14%	0.00%	2.70%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
335	0.14%	0.14%	0.14%	0.14%	0.00%	6.48%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
336	0.14%	0.14%	0.14%	0.14%	0.00%	12.10%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
337	0.14%	0.14%	0.14%	0.14%	0.00%	17.60%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
338	0.14%	0.14%	0.14%	0.14%	0.00%	19.95%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
339	0.14%	0.14%	0.14%	0.14%	0.00%	17.60%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
340	0.14%	0.14%	0.14%	0.14%	0.00%	12.10%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
341	0.14%	0.14%	0.14%	0.14%	0.00%	6.48%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
342	0.14%	0.14%	0.14%	0.14%	0.00%	2.70%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
343	0.14%	0.14%	0.14%	0.14%	0.00%	0.88%	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%
564	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.88%	0.00%
565	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	2.70%	0.00%
566	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	6.48%	0.00%
567	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	12.10%	0.00%
568	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	17.60%	0.00%
569	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	19.95%	0.00%
570	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	17.60%	0.00%
571	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	12.10%	0.00%
572	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	6.48%	0.00%
573	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	2.70%	0.00%
574	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.88%	0.00%
687	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	0.88%
688	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	2.70%
689	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	6.48%
690	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	12.10%
691	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	17.60%
692	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	19.95%
693	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	17.60%
694	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	12.10%
695	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	6.48%
696	0.14%	0.14%	0.14%	0.14%	0.00%	0.00%	0.14%	0.14%	0.14%	0.14%	0.00%	2.70%

Figure 90: Un exemple général d'une matrice de confiance. On a associé les deux segments similaires (207→338 et 569→692) aux deux instructions similaire (t5-t6 et t11-t12). Sur les autres colonnes, pour lesquelles cet outil ne fourni aucune information, le coefficient est distribué d'une manière équiprobable sur toutes les lignes. Le fait de remplir ces colonnes par des zéros signifirait que l'outil est sûr que cette colonne n'a aucune correspondance parmi les lignes, ce qui n'est pas le cas (voir la section 5.2.3 – la somme des coefficients sur chaque colonne est toujours égal à 1).

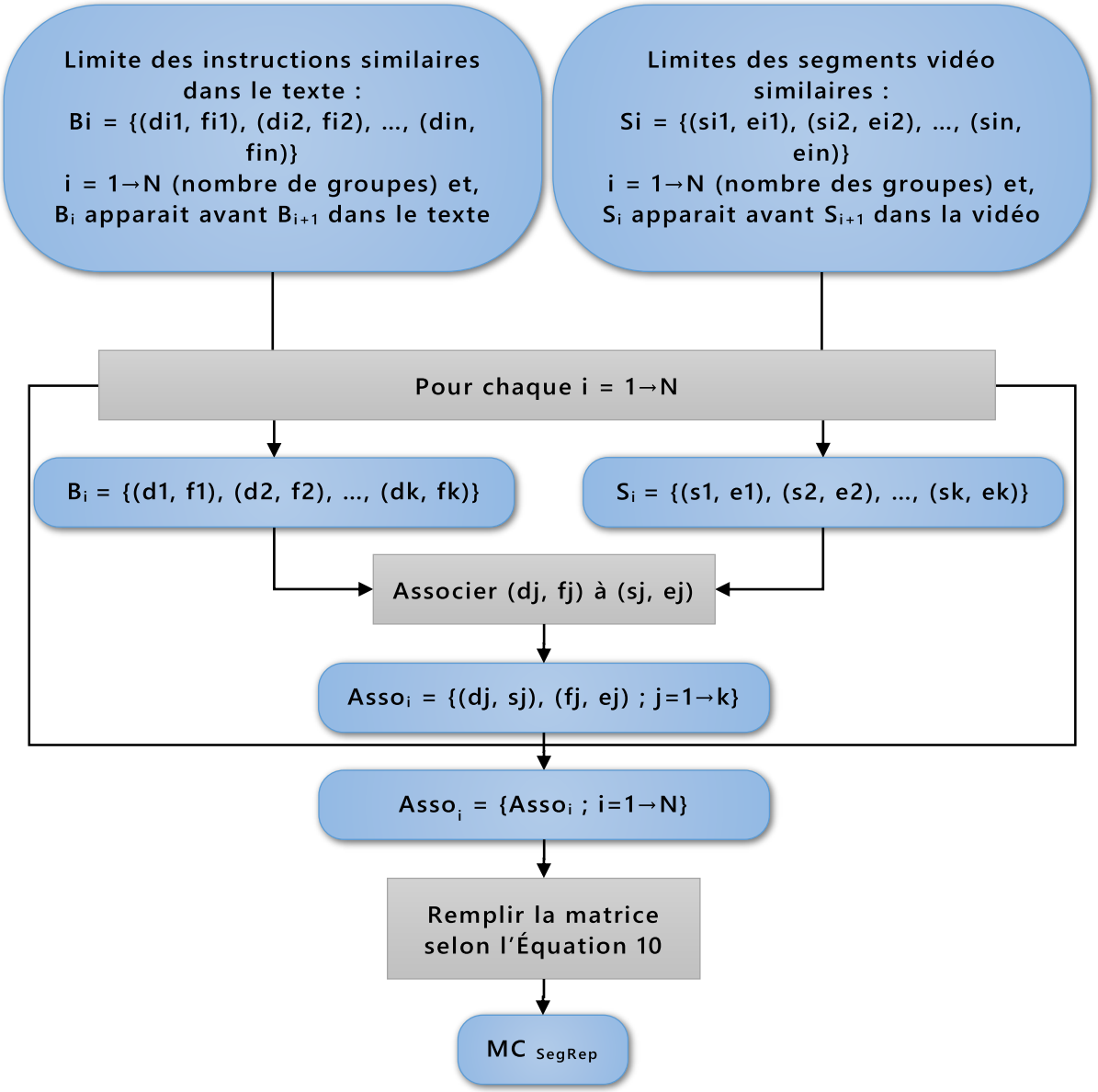


Figure 91 : Remplissage de la matrice de confiance MC de l'outil de détection des segments répétés.

5.3 Fusion des matrices

Nous avons maintenant un ensemble de matrices de confiance calculées par plusieurs outils en nous basant sur des informations extraites automatiquement. Cet ensemble est extensible de manière à pouvoir intégrer n'importe quel autre outil capable d'extraire d'autres informations pertinentes sur la vidéo en rapport avec le texte. La fusion de ces matrices dans une matrice unique doit maintenant nous permettre d'extraire les associations finales entre les limites.

Par exemple, considérons une répétition qui consiste à extraire 7 objets à partir d'une boite et à les déposer sur une table. Cette répétition est détectée principalement par l'outil de détection des répétitions en boucle. Le début (resp. la fin) du segment est

alors associé au début (resp. à la fin) de la boucle dans le texte. Comme les occurrences de cette répétition modifient la scène originale (un objet est ajouté à chaque fois sur la table), l'outil de segmentation d'actions détecte aussi les limites de ces occurrences, y compris le début (début de la première occurrence) et la fin (fin de la dernière occurrence) de la répétition. Donc, le début et la fin de la boucle sont associés aux mêmes limites textuelles par les deux outils de manière indépendante mais – très probablement – avec quelques trames d'écart. Dans ce cas, nous attendons de la fusion des matrices de produire de nouveaux coefficients dans les colonnes "début" et "fin" de la boucle marquant la cohérence des résultats des deux outils malgré une possible imprécision sur la position temporelle.

5.3.1 Fusion pondérée

La fusion de plusieurs informations provenant d'outils ayant des fiabilités différentes pose le problème de la prise en compte de la différence des fiabilités. Nous proposons d'associer à chaque matrice un coefficient qui reflète le niveau de la fiabilité de l'outil qui l'a produite. À partir de là, nous effectuons une fusion pondérée des matrices de la façon suivante :

$$MC_{Finale} = \sum_{k=1}^{N} a_k \times MC_k \,; \qquad tel\ que: \sum_{k=1}^{N} a_k = 1$$

Équation 30

Où :
- → $\mathbf{MC_k}$ est la matrice de confiance calculée par l'outil **k**,
- → $\mathbf{a_k}$ est le coefficient de fiabilité de la matrice $\mathbf{MC_k}$
- → **N** est le nombre d'outils exploités

Le coefficient $\mathbf{a_k}$ d'un outil **k** est une valeur calculée en fonction de la fiabilité de l'outil. Comme le calcul de la fiabilité de l'outil est compliqué et dépend souvent de plusieurs facteurs, le calcul d'une valeur fixe d'$\mathbf{a_k}$ applicable dans tous les cas, est plus complexe. Dans notre étude, nous proposons de quantifier cette fiabilité en nous référant à des éléments de résultats obtenus sur un corpus significatif.

Dans ce qui précède, nous remarquons que le paramètre **σ** est une mesure qui reflète la fiabilité des limites automatiques sélectionnées. Plus ce paramètre est petit, plus l'outil est fiable. De ce fait, nous proposons d'utiliser la formule suivante pour calculer la fiabilité d'un outil :

$$Fiabilité_k = {}^{1}/_{\sigma_k} \,;$$

Équation 31

Où :
- → $\mathbf{\sigma_i}$ est l'écart-type des distances entre les limites automatiques et les limites de la vérité terrain, selon l'outil **k**.

D'où la fiabilité de chaque outil :

Outil de segmentation :

$$\textbf{Fiabilité}_1 = 1 / 40.6662 = 0.0234$$

Outil des répétitions :

$$\textbf{Fiabilité}_2 = 1 / 37.5367 = 0.0266$$

Et les coefficients normalisés :

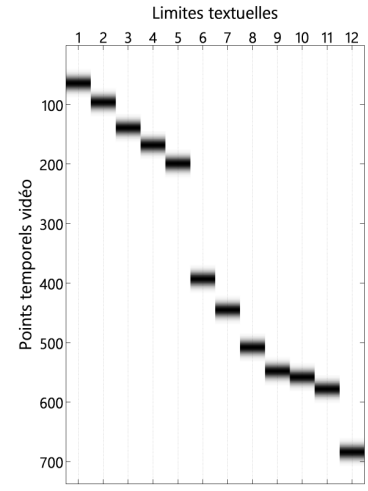
$$a_k = {Fiabilité_k} \Big/ {\sum_{i=1}^{n} Fiabilité_i} ;$$

où n est le nombre des outils.

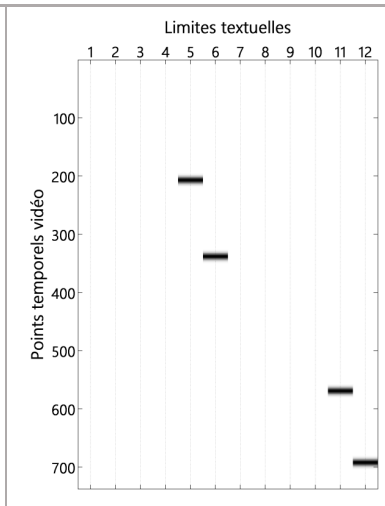
Outil de segmentation : $\mathbf{a_1 = 0.468}$
Outil des répétitions : $\mathbf{a_2 = 0.532}$

Équation 32

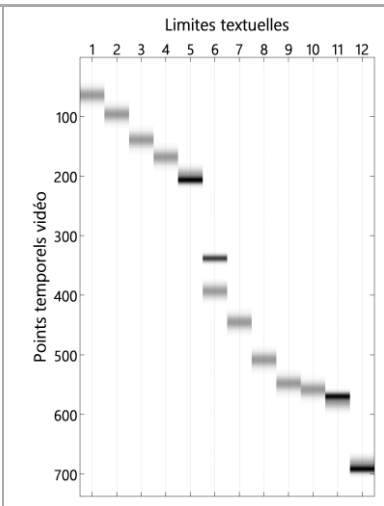
Pour montrer l'effet du choix d'$\mathbf{a_k}$ sur la qualité des résultats, nous présenterons plus tard durant l'évaluation du système, les résultats d'évaluation en utilisant des différentes combinaisons de coefficients $\mathbf{a_k}$ (cf. Figure 99). Dans cette figure, nous remarquons que les indices déjà calculés fournissent le deuxième meilleur résultat par comparaison avec les autres combinaisons.



(a) Matrice de confiance de l'outil de segmentation d'actions.



(b) Matrice de confiance de l'outil de détection des répétitions en boucle.



(c) La matrice finale de confiance calculée en fusionnant les deux matrices (a) et (b), selon l'Équation 30

Figure 92 : les matrices de confiances d'une vidéo réelle formée de 12 limites, y compris deux boucles (section 5.2.3.2.2).

5.4 Extraction des associations

5.4.1 Extraction simple

À ce point, les informations fournies par tous les outils sont combinées dans une matrice finale de confiance. Dans cette matrice, l'entrée ayant le coefficient le plus élevé correspond à l'association "limite vidéo ↔ limite textuelle" la plus fiable.

Nous avons déjà indiqué que ce système considère les limites textuelles en tant que référence pour associer les limites vidéo correspondantes. Ce fait est traduit dans la matrice de confiance par une somme de coefficients égale à 1 dans chaque colonne, ce qui signifie que chaque limite textuelle a forcément une correspondance dans les limites vidéo (voir la section 5.2.3). Dans cette matrice, la ligne contenant le coefficient le plus élevé est considéré comme étant la limite vidéo qui correspond à cette limite textuelle. Par suite, l'identification de cette ligne pour chaque colonne devrait permettre de localiser dans la vidéo les limites textuelles.

La solution la plus simple est de chercher d'une manière indépendante la valeur maximale dans chaque colonne, et de lui associer la ligne correspondante. Mais la matrice finale est le résultat de la combinaison d'informations fournies par des outils indépendants, ce qui peut conduire à des contradictions. Par exemple, la fin d'une instruction textuelle peut être détectée avant son début. De ce fait, l'extraction des associations de la matrice finale doit être effectuée en respect d'un ensemble des contraintes.

5.4.2 Association récursive

Nous proposons une extraction récursive au lieu d'une extraction simple sans contraintes, pour deux raisons :

- → Dans cette solution, nous prenons en considération l'ordre des limites textuelles et vidéo,
- → Après l'extraction de chaque association, les futures zones d'association possibles sont mises à jour pour contraindre le champ des possibles. Cette mise à jour permet de faire apparaître la contribution de nouveaux points temporels moins fiables, mais plus cohérents avec les associations précédemment sélectionnées.

On propose la fonction récursive suivante pour extraire les associations d'une matrice finale de confiance **MC** donnée :

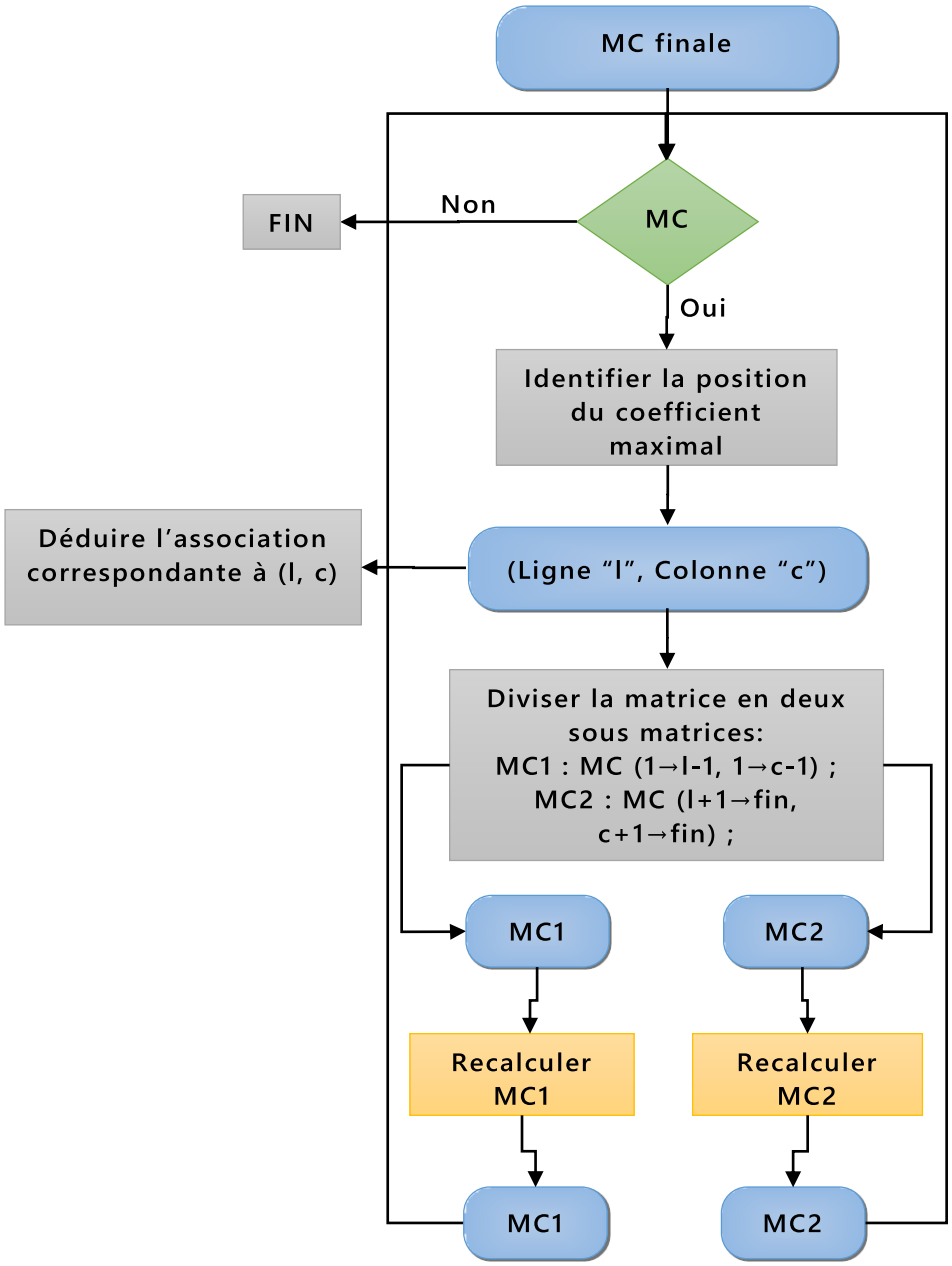


Figure 93 : procédure récursive d'extraction des associations.

Dans la Figure 93, la fonction "Recalculer MC" consiste à reprendre la procédure de la Figure 74, mais sans relancer les outils. L'objectif ici est de recalculer les matrices MC de chaque outil.

Le fait de relancer l'extraction des limites/segments pour le calcul des sous-matrices permet potentiellement de faire remonter des points temporels non sélectionnés dans un premier temps (lors de la sélection des **N** limites les plus significatives en regard des **N** limites textuelles). Durant le premier remplissage de la matrice d'un outil de type 1, on sélectionne un nombre de points temporels égal au nombre de limites textuelles à associer. Ces points ont les votes les plus élevés. Si nous n'utilisons que ces points durant l'association, nous pouvons obtenir des sous-matrices

dans lesquelles aucun point marquant une limite vidéo n'est présent. Par contre, en sélectionnant un nouvel ensemble de points sur chaque intervalle sur lesquels la fonction d'association est appelée de manière récursive, nous aurons toujours des points temporels candidats. Par exemple, un point temporel qui n'était pas parmi les **N** plus forts points sur l'intervalle complet de la vidéo (Figure 94) peut être sélectionné parmi les **M** (<**N**) plus forts points au niveau d'un sous-intervalle (Figure 95).

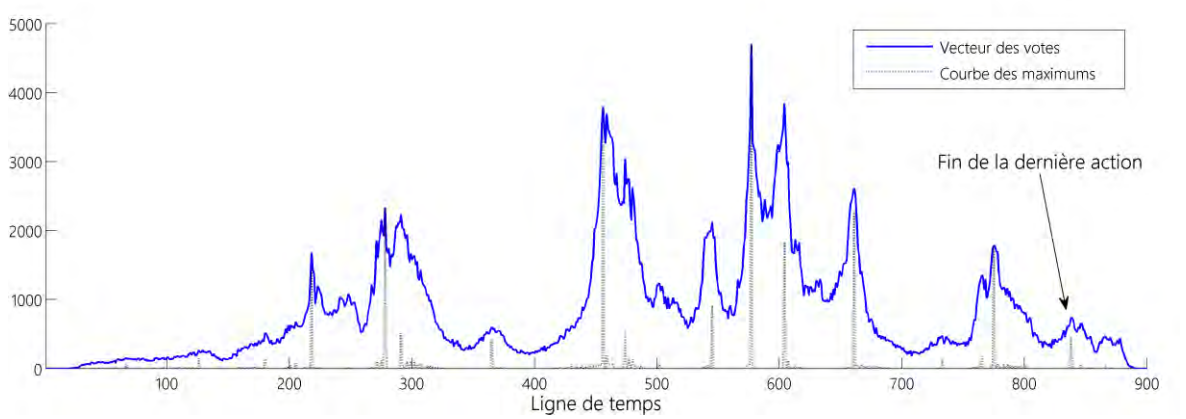


Figure 94: Un exemple d'un point temporel (qui représente la fin de la dernière action) ayant un vote bas au niveau de la vidéo entière. Au début de la procédure d'association des segments/limites, ce point n'est pas sélectionné parmi les points ayant les 8 (nombre des limites textuelles dans cette vidéo) votes les plus élevés. Il n'est donc pas associé à une limite textuelle point à point lors de la première itération de l'algorithme d'association.

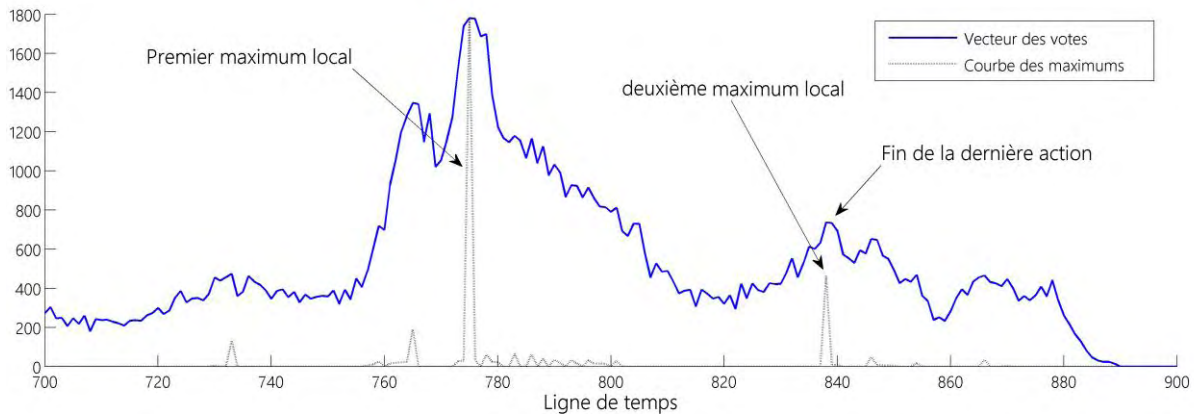


Figure 95: Durant le re-calcul d'un sous-intervalle obtenu après plusieurs appels récursifs, dans lequel nous cherchons les correspondances de deux limites textuelles, ce point est sélectionné parmi les deux votes les plus élevés. Il est alors associé à la limite correspondante.

5.5 Résultats et évaluation

5.5.1 Le corpus

La synchronisation d'un contenu textuel et d'un contenu vidéo (en dehors de l'alignement de script) est un sujet peu abordé dans le domaine de l'analyse des

contenus multimédia. Ceci justifie l'absence des grands corpus permettant d'évaluer correctement notre système. En plus d'un corpus formé des couples texte-vidéo, où chaque texte décrit la démarche suivie dans l'enregistrement vidéo sous forme d'une séquence d'instructions, nous avons besoin d'annotations précises permettant d'évaluer les résultats automatiques du système. Selon nos informations, il n'existe pas de tels corpus.

Les corpus de la littérature sont conçus et construits pour tester et évaluer les méthodes qui traitent les problèmes les plus populaires dans le domaine : reconnaissance des gestes ((38), (39)), extraction d'arrière-plan ((40), (41)), structuration et analyse du flux télévisé (chaînes de télévision), etc... Aucun de ces corpus n'est éligible pour être utilisé pour évaluer notre système de synchronisation.

Les corpus qui peuvent coïncider avec nos besoins sont ceux qui contiennent des recettes de cuisine et leurs enregistrements vidéos ou ceux qui contiennent des tutoriels pour la réalisation d'une tache spécifique (maintenance, assemblage de produits en kit, *etc*...) et les vidéos qui montrent quelqu'un réalisant cette tâche. Ce type de contenus se trouve sur de nombreux sites internet spécialisés. La principale limitation pour leur utilisation est que ces contenus vidéo sont souvent édités et ont subi des opérations de montage. Les producteurs utilisent plusieurs caméras parfois mobiles, des effets d'illumination, des changements complets de la scène (arrière-plan), et ainsi de suite. Ceci ne correspond pas à nos hypothèses de notre travail. Selon les hypothèses des outils utilisés, nous considérons des vidéos créées par une seule caméra stationnaire.

Nous avons donc créé notre propre corpus dont la taille est juste suffisante pour tester et vérifier les algorithmes et les méthodes proposées, ainsi que pour présenter des échantillons de résultats obtenus.

Ce corpus consiste en des vidéos dans lesquelles un opérateur humain effectue des actions qui affectent la scène originale, conformément aux instructions rédigées dans des textes prédéfinis. Le corpus est décrit dans le tableau suivant :

Informations générales		
Nombre des vidéos qui contiennent des actions unitaires et des répétitions en boucle	11	vidéos
Nombre des vidéos qui ne contiennent que des actions unitaires	2	vidéos
Durée minimale des vidéos	470	trames
Durée maximale des vidéos	2710	trames
Durée totale des vidéos	16378	trames

Durée totale des segments à détecter[7] (Actions + répétitions)	7906	trames

Actions unitaires		
Nombre d'actions unitaires	98	actions unitaires
Durée minimale d'une action	17	trames
Durée maximale d'une action	105	trames
Nombre d'actions par vidéo	3 → 13	actions
Durée totale des actions unitaires	4515	trames

Répétitions		
Nombre total de répétitions	20	répétitions
Nombre de répétitions dans une vidéo	1→3	répétitions
Durée minimale d'une répétition	110	trames
Durée maximale d'une répétition	866	trames
Durée totale des répétitions	6496	trames

Tableau 21: Détails du corpus utilisé.

Toutes les vidéos sont enregistrées en utilisant la même caméra Nikon D5100 installée sur un trépied. Les réglages sont effectués de façon à minimiser le bruit. Toutes les vidéos sont au même format et ont les mêmes dimensions 320x180 pixels (choisies de manière à induire des temps de calcul raisonnables dans nos conditions expérimentales) Les trames sont extraites en utilisant l'outil "**ffmpeg**" avec ses paramètres configurés par défaut. Les vidéos sont enregistrées sous différents type de source d'illumination de manière séparée : fluorescent, incandescent, lumière naturelle à l'intérieur et lumière naturelle à l'extérieur.

Les actions unitaires consistent à effectuer une modification sur la scène originale (*ex*. déplacer et ajouter des objets), et les répétitions consistent à répéter soit des actions qui agissent sur la scène originale (*ex*. déplacer plusieurs objets de la même façon), soit des actions qui n'agissent pas sur la scène (*ex*. sauter plusieurs fois). Les sujets des vidéos sont :

[7] Ce n'est pas la somme des durées totales des répétitions et des actions, car il y a des segments communs. Pour une répétition qui consiste à effectuer N modifications sur l'arrière-plan, le système détecte le N actions unitaires et la répétition entière.

1) *Déplacement d'objets :*

   a. *Actions :* *déplacer des objets dans la scène, de l'extérieur et vers l'intérieur*

   b. *Répétition :* *déplacer plusieurs objets de la même façon ou répéter un mouvement (ex. sauter).*

2) *Recette de cuisine :*

   a. *Actions :* *ajouter/déplacer des ingrédients à/dans la scène,*

   b. *Répétition :* *faire une action répétitive (ex. ajouter plusieurs cuillères d'un ingrédient, dissoudre du sucre en tournant une cuillère).*

3) *Jeux des cartes (distribution des cartes) :*

   a. *Actions :* *déposer les cartes devant les joueurs, couper un paquet de cartes,*

   b. *Répétition :* *distribuer des cartes à chaque joueur.*

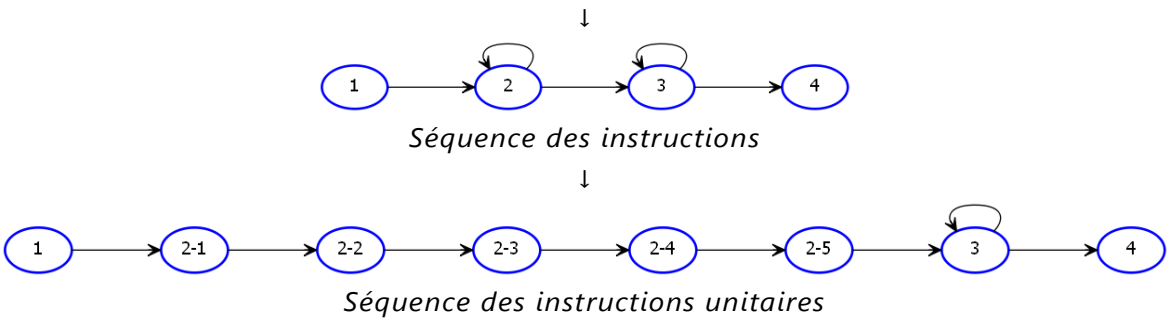
4) *Réalisation de la forme d'un bus à partir d'un ensemble de pièces :*

   a. *Actions :* *mettre chaque pièce à sa position pour former une forme de "Bus",*

   b. *Répétition :* *effectuer des actions similaires, ex. placer les cinq pièces figurant les fenêtres aux places convenables.*

Les instructions textuelles

1) **Ajouter une chaise,**
2) **Approcher les 5 objets à gauche vers la caméra,**
3) **S'asseoir sur la chaise et se relever 7 fois,**
4) **Déplacer la chaise vers la droite.**



Séquence des instructions

Séquence des instructions unitaires

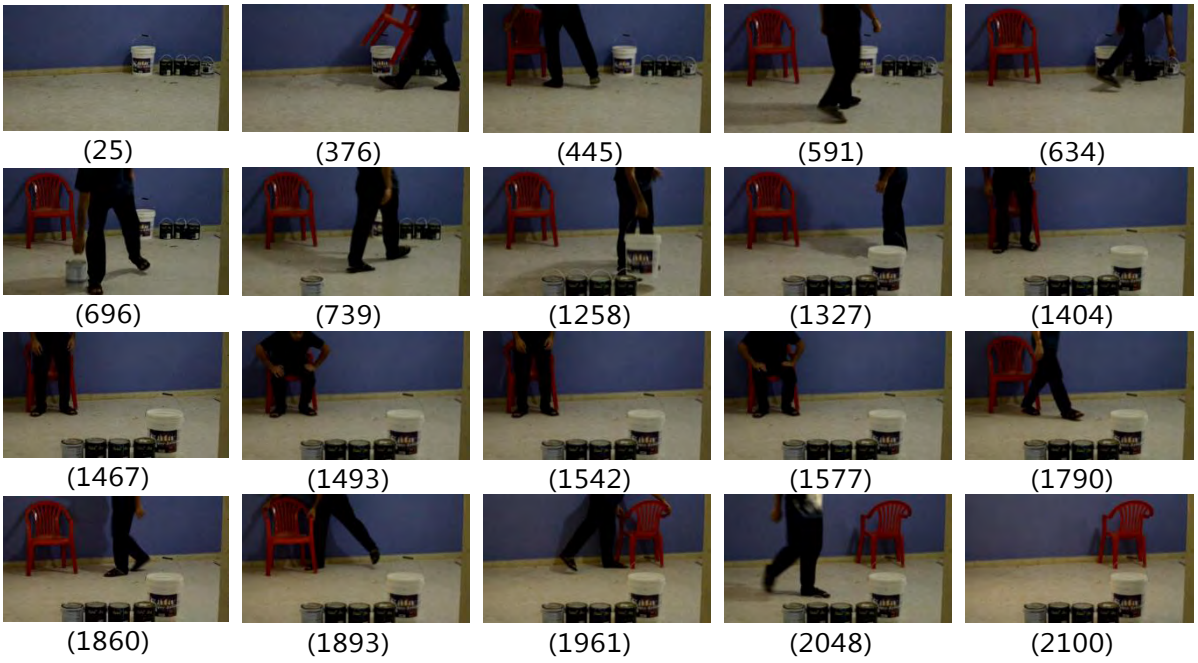
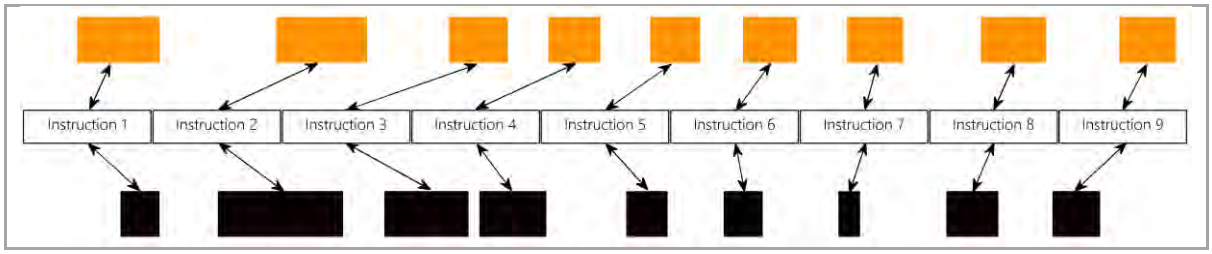


Figure 96: Exemple d'un contenu textuel et des échantillons du contenu vidéo correspondant. Dans cette vidéo, un opérateur humain exerce des actions dans son environnement : ajoute un objet à la scène originale (trames : 376→445), déplace plusieurs objets de la même manière (590→1258), s'assoit et se relève de la chaise plusieurs fois (1404→1577), déplace un objet à l'intérieur de la scène (1860→1961).

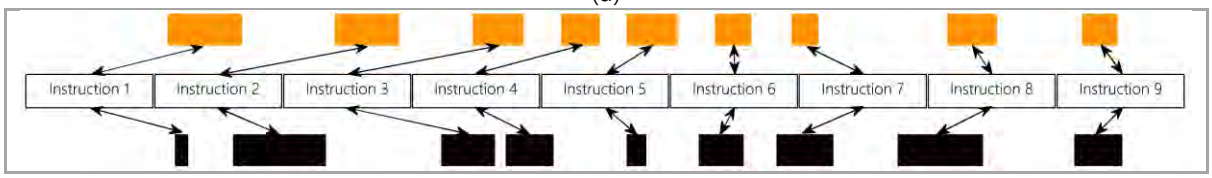
5.5.2 Résultats

On présente dans cette section, les résultats de synchronisation entre les contenus textuels et les contenus vidéo correspondants dans le corpus créé. Dans le calcul de ces résultats, nous nous sommes appuyés sur les hypothèses formulées à propos de l'analyse de texte. Nous avons ainsi supposé qu'il existe des outils d'analyse textuelle fiables qui fournissent une séquence d'actions unitaires, ainsi que d'autres informations comme le nombre de boucles et l'existence d'instructions similaires (*cf.* le Chapitre 2 – section 2.4).

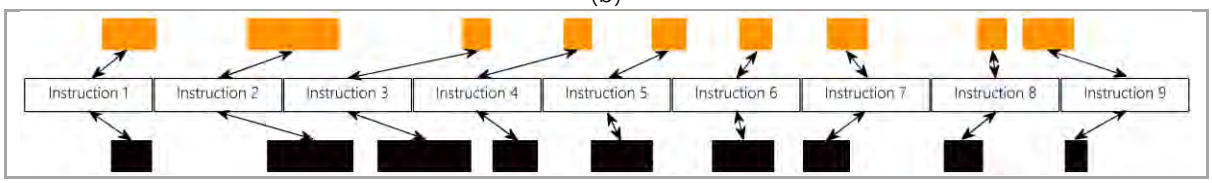
Nous avons effectué le processus de synchronisation en n'utilisant que la segmentation des actions et la détection des répétitions. Nous n'avons pas utilisé l'outil de détection des actions similaires car nous n'en disposions pas d'une version exploitable pour cette étude. En nous basant sur les informations produites, nous calculons la matrice de confiance de chaque outil, et nous les fusionnons en une seule matrice finale. Finalement, nous appliquons la méthode récursive pour extraire les associations des limites.



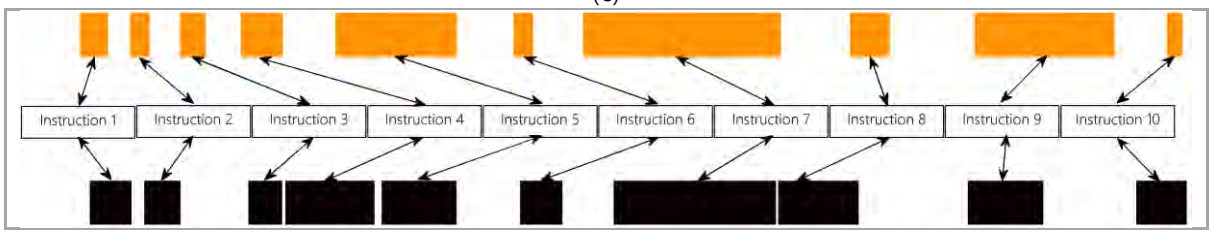
(a)



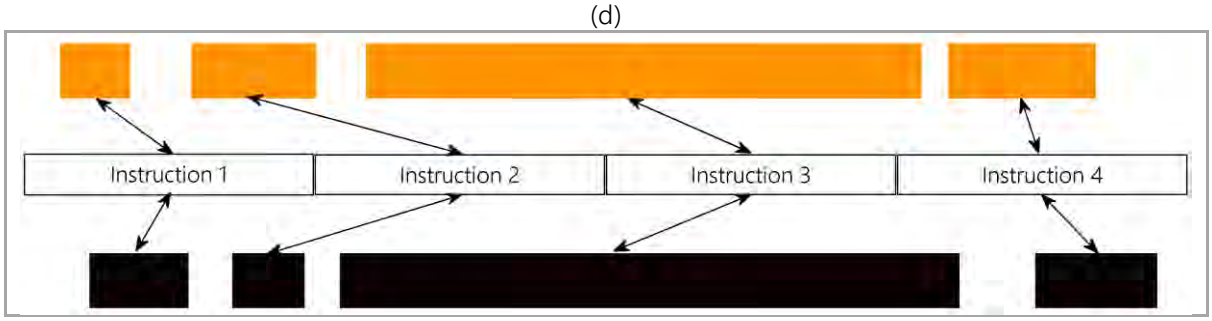
(b)



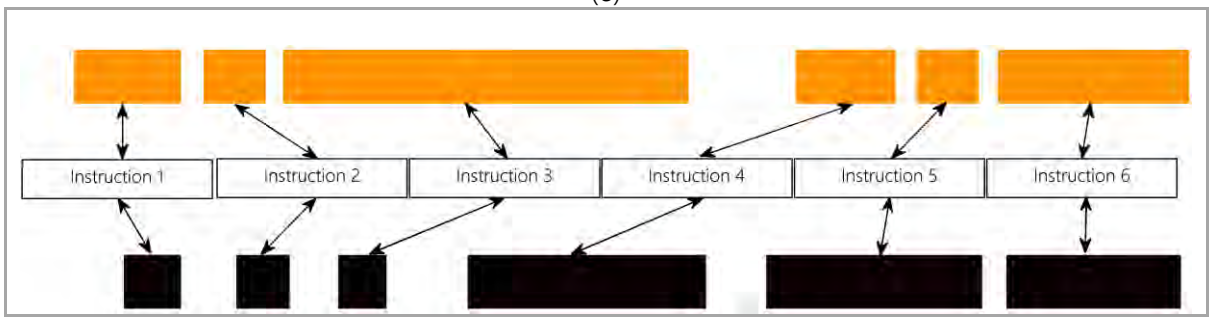
(c)



(d)



(e)



(f)

Figure 97 : les résultats de synchronisation de six enregistrements vidéo avec leurs contenus textuels correspondants. L'axe horizontal correspond à la ligne de temps et un rectangle représente un segment vidéo sur cette ligne. Dans chaque figure, les cibles entre les instructions textuelles et les segments orange (resp. noirs) montrent les associations entre la liste d'instructions et les segments annotés manuellement : "vérité terrain" (resp. automatiquement : "notre système").

Dans ces figures, nous désignons par une association erronée, une association d'une instruction à un segment qui :

→ ne contient pas sa réalisation comme l'association de l'instruction 3 dans la figure (c) : le segment associé à cette instruction ne contient aucune réalisation d'instruction.

→ contient la réalisation d'une instruction voisine comme l'association de l'instruction 3 dans la figure (d) : le segment associé à cette instruction contient la réalisation de l'instruction 4.

Selon cet échantillon, nous remarquons que notre système associe les instructions textuelles aux segments vidéo correspondants, parfois d'une manière fiable (ex. l'association de l'instruction 6 dans la figure (f)), et d'autres fois avec une erreur marquée (ex. l'association de l'instruction 4 dans la vidéo (f)).

Dans la section suivante, nous proposons des méthodes pour évaluer le système proposé. Ces méthodes étudient les résultats obtenus et les comparent à la vérité terrain pour fournir des valeurs statistiques qui donnent une idée concernant la qualité des résultats obtenus en regard du volume des données étudiées.

5.5.3 Évaluation

Nous proposons ici d'évaluer la capacité du système à associer un segment vidéo à l'instruction textuelle correspondante. La précision de la délimitation des segments par rapport à la vérité terrain ne sera pas prise en compte. La segmentation a déjà été évaluée dans les chapitres précédents (outils de segmentation et de détection des répétitions).

Notre système propose des associations entre des segments vidéo et des instructions textuelles. Ces associations doivent être comparées aux associations entre les segments vidéo de la vérité terrain et les instructions textuelles.

Étant donnée une association automatique segment ↔ instruction SA↔LA, cette association est considérée correcte s'il existe une association manuelle (vérité terrain) SM↔LM telle que :

→ **LA ≡ LM et,**

→ **SA (resp. SM) est le segment automatique (resp. manuel) qui couvre la plus grande partie de SM (resp. SA).**

Instructions	Segments automatiques		Segments manuels	
Instruction 1	SA1 :	A1 → A1+ α1	SM1 :	M1 → M1 + β1
Instruction 2	SA2 :	A2 → A2 + α2	SM2 :	M2 → M2 + β2
Instruction 3	SA3 :	A3 → A3 + α3	SM3 :	M3 → M3 + β3
…		…		…
Instruction n	SAn :	An → An + αn	SMn :	Mn → Mn + βn

Tableau 22 : Le segment associé automatiquement et celui associé manuellement, à chaque instruction textuelle où, α_i (resp. β_i) est la taille du segment qui correspondant à l'instruction i selon notre système (resp. selon la segmentation manuelle).

Nous considérons que l'instruction **i** est bien associée par notre système, si :

$$(SA_i \cap SM_i) \neq \emptyset \text{ et,}$$
$$(SA_i \cap SM_i) > \{SA_i \cap SM_j\,;\, j=1\rightarrow n,\, j \neq i\} \text{ et,} \qquad \text{Équation 33}$$
$$(SA_i \cap SM_i) > \{SA_j \cap SM_i\,;\, j=1\rightarrow n,\, j \neq i\}$$

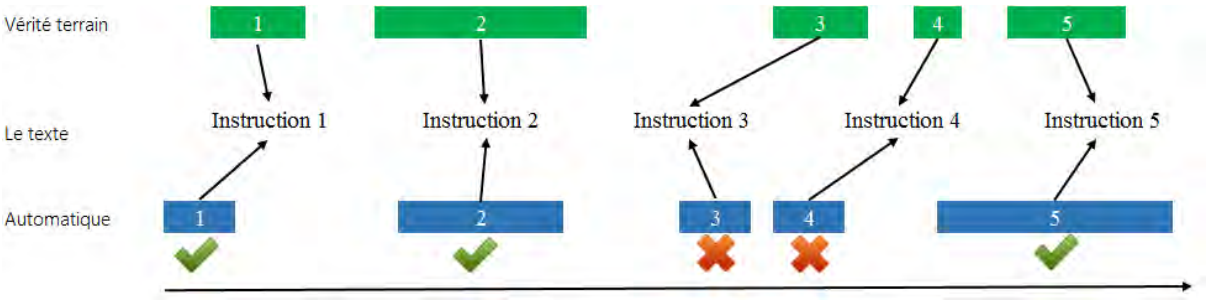


Figure 98 : Les associations correctes et les associations erronées. A l'intérieur de chaque segment on indique le numéro d'instruction y associé.

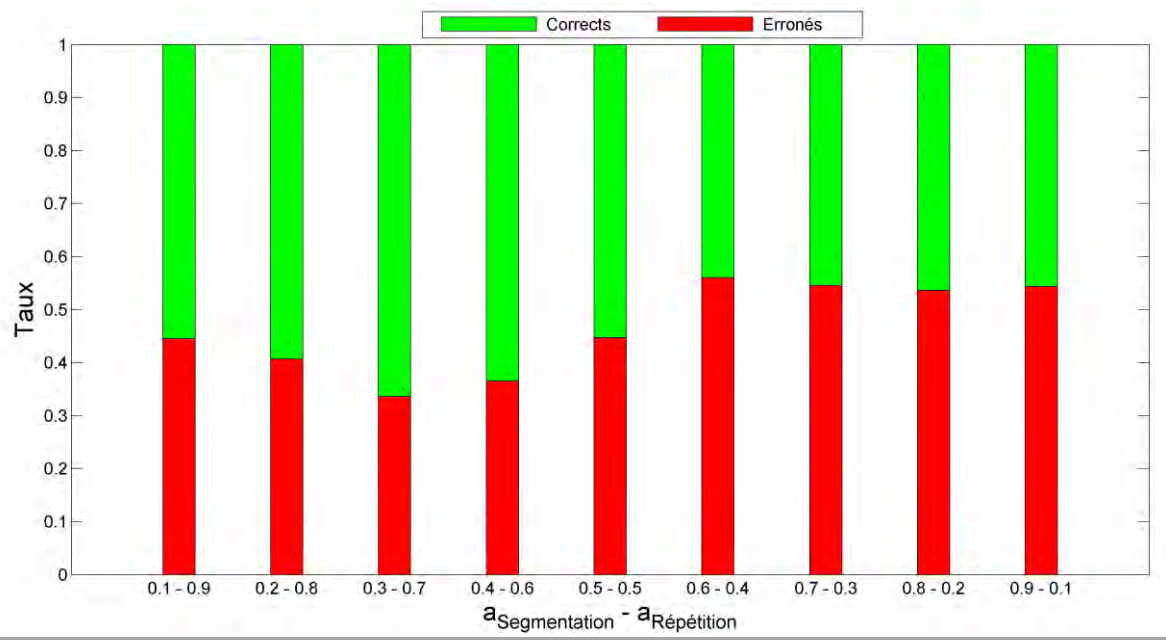
Dans l'exemple ci-dessus, notre système propose 5 associations "instruction ↔ segment". Ces associations sont extraites de la matrice finale de confiance à l'aide de notre méthode récursive. De même, la segmentation et l'association manuelles fournissent 5 associations "instruction ↔ segment" (Tableau 22). À ce point, si l'intersection entre les segments de l'instruction i dans le Tableau 22 (**SA**$_i$ et **SM**$_i$) vérifie l'Équation 33 , alors l'association de cette instruction est considérée correcte. Dans l'exemple de la Figure 98, trois associations sont correctes et deux sont erronées.

Nous proposons deux taux d'erreur qui peuvent être calculés pour cette vidéo :

→ Taux d'associations correctes :
(Nombre d'associations correctes) / (Nombre d'associations)

→ Taux d'associations erronées :
(Nombre des associations erronées) / (Nombre des associations) = 1 – (Taux d'associations correctes)

Ensuite, nous calculons les moyennes des taux calculés sur chaque vidéo pour avoir les taux globaux. Nous présentons les résultats obtenus pour différentes valeurs des coefficients de pondération :



Corrects								
0.55	0.59	0.66	0.63	0.55	0.44	0.45	0.46	0.46
Erronés								
0.45	0.41	0.34	0.37	0.45	0.56	0.55	0.54	0.54

Figure 99 : Taux calculés sur le corpus décrit dans le Tableau 21.

Dans le Figure 99, nous présentons le taux des associations correctes et le taux des associations erronées. Nous remarquons aussi que les coefficients $\mathbf{a}_k$ qui minimisent les associations erronées sont ceux qui donnent plus d'importance à la matrice des répétitions ($\mathbf{a}_{Segmentation}$=0.3 et $\mathbf{a}_{Répétition}$=0.7), ce qui indique que le deuxième outil fournit des informations plus fiables que le premier. Ces coefficients ne sont pas très éloignés des coefficients déjà calculés dans la section "5.3.1 : Fusion pondérée", ce qui confirme que l'utilisation de l'écart-type comme mesure de fiabilité, est un choix raisonnable.

Nous remarquons, comme nous pouvons nous y attendre, qu'une association détectée d'une manière erronée affecte toutes les associations suivantes dans la vidéo. Dans la Figure 98, le "segment automatique" numéro 4 qui contient vraiment l'exécution de l'instruction 3, est associé à l'instruction 4 car le segment numéro 3 qui ne contient vraiment aucune action (segment mal détecté par un des outils) est associé à l'instruction 3. Dans ce cas, cette erreur va induire un décalage jusqu'à la fin de la vidéo sauf si une autre erreur annule la première. Bien que le segment numéro 4 contienne une exécution correcte (celle de l'instruction 3), l'association correspondante

est considérée comme erronée. Pour étudier cette possibilité de décalage, on calcule les taux des associations décalées d'une et deux instructions. De cette manière, l'association du segment numéro 4 de la Figure 98 pourra être comptée comme étant correcte à une instruction près.

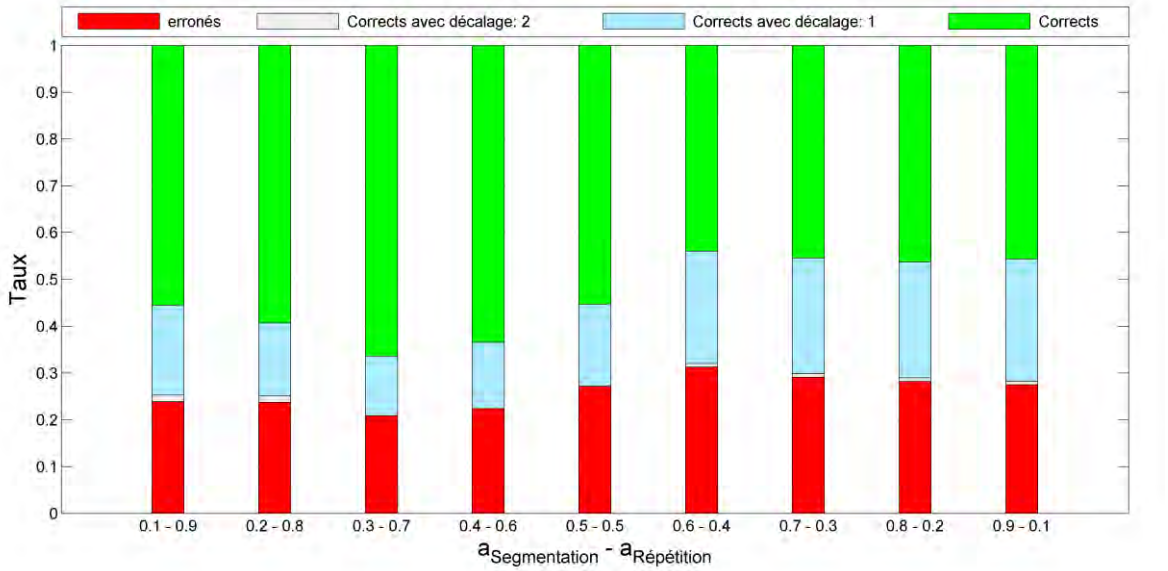


Figure 100 : Identique à la Figure 99, mais en distinguant les taux des associations décalées.

On remarque que les taux des associations subissant un décalage d'une instruction varient entre 16 % et 29 % selon les coefficients de fusion utilisés. Ceci signifie que les segments sont correctement détectés mais mal associés à cause d'une erreur propagée.

Chapitre 6

Conclusion et Perspectives

Dans le cadre de ce travail, nous avons abordé le problème de la synchronisation entre une vidéo et un texte qui décrit son contenu. Une des originalités de ce travail consiste en la mise en correspondance entre un contenu temporel (vidéo) et un autre non-temporel (le texte). Le système proposé est un système générique capable d'intégrer plusieurs types d'informations et de les fusionner pour produire des éléments de synchronisation.

Nous avons proposé deux outils d'analyse vidéo qui extraient deux types différents d'informations. Chaque outil traite la vidéo en cherchant d'un type d'information prédéfini : le premier extrait les limites des actions effectuées et l'autre détecte les segments qui contiennent une action répétée. Les limites fournies par le premier outil sont associées aux limites des instructions unitaires dans le texte. Le deuxième repère les segments qui peuvent correspondre aux instructions qui exigent l'exécution d'une répétition (une boucle). Ces outils ne s'intéressent pas à la reconnaissance en tant que telle des actions exécutées, mais sont conçus pour extraire les limites des actions tant qu'elles respectent le modèle du type de segment cherché.

Par exemple, l'outil de détection des répétitions trouve les segments qui contiennent n'importe quelle action du moment qu'elle est répétitive.

Dans la deuxième partie, nous avons proposé une représentation matricielle construite à partir des résultats de chaque outil pour identifier leurs associations potentielles avec les instructions textuelles sous forme de coefficients de confiance. Ces coefficients sont calculés en fonction de la fiabilité de l'outil. Les matrices ainsi construites pour chaque outil sont ensuite fusionnées en une matrice finale de confiance. C'est à partir de cette matrice qu'une méthode récursive extrait les associations finales entre chaque segment vidéo et l'instruction textuelle correspondante.

La méthode de synchronisation se veut indépendante des algorithmes de segmentation, et même des types d'action analysés. Cela doit permettre d'intégrer à terme un grand nombre d'outils capables de fournir différents types d'information. Cela pourrait permettre de rendre le résultat du système plus fiable et plus précis.

Bien évidemment, à ce point, nous ne prétendons pas que ce système forme une solution complète au problème posé, malgré les résultats encourageants obtenus. Mais nous avons confiance dans le fait que ce système soit une base pour des futurs travaux qui doivent intégrer plus d'outils et d'idées pour améliorer la qualité des résultats.

Nous donnons dans les paragraphes suivants quelques perspectives qui nous semblent prometteuses.

6.1 Outil de détection des répétitions

Problématique :

Nous avons proposé un outil de détection des répétitions en boucle dans une vidéo. Cet outil représente la vidéo sous forme d'un signal dont les répétitions apparaissent comme des segments périodiques. Cette méthode de représentation fonctionne bien dans des conditions respectant les hypothèses de notre travail. Or, ce n'est pas toujours le cas en dehors de ce contexte. Donc pour généraliser l'utilisation de l'outil, une méthode avancée de représentation doit être proposée.

Proposition :

Nous proposons de créer des signaux s'appuyant sur plusieurs caractéristiques vidéo comme le flot optique, l'orientation du mouvement dominant, la saturation des couleurs, etc.... Ensuite, une méthode de fusion précoce pourrait être appliquée pour unifier les signaux calculés en un seul signal. À ce point, ce signal qui représenterait bien les répétitions par des segments périodiques, pourrait alimenter notre méthode

de détection des répétitions (Chapitre 4 : Détection des Répétitions) sans modification majeure.

6.2 Outil de détection des répétitions – coût de calcul

Problématique et proposition:

Dans certains cas, le contenu textuel peut contenir des informations temporelles qui fournissent par exemple le temps nécessaire pour réaliser la répétition. Ces informations qui peuvent être extraites par l'outil textuel, apparaissent sous forme d'une estimation (ex. "taper 10 fois sur la table [est. 3 à 7 secondes]") ou d'une période fixe (ex. "maintenez à ébullition tout en mélangeant pendant 30 secondes"). Ce type d'information est utile pour affiner les limites des segments qui contiennent une répétition, et pour minimiser le coût de calcul. Étant donnée la durée D d'une répétition :

→ MATYIN : Nous calculons la partie qui contient uniquement les triangles correspondants aux répétitions dont leurs durées sont dans un intervalle donné. Cette partie est limitée par deux lignes (dans le MATYIN, ligne ≡ taille de segment) : [D - ε, D + ε],

→ ANGLEDROIT : Nous cherchons uniquement les angles droits situés entre les deux lignes : (D – ε) et (D + ε).

6.3 Le flux audio

Problématique :

Dans notre travail, nous nous sommes concentrés uniquement sur l'analyse vidéo pour réaliser la synchronisation des contenus. En général, une vidéo est accompagnée par un flux audio qui pourrait être utile pour la synchronisation.

Proposition :

Parfois le flux audio est complétement indépendant du contenu visuel, parfois il est lié mais inexploitable comme source d'information, et parfois encore il n'est tout simplement pas disponible. Mais on peut supposer que le signal audio accompagnant la vidéo peut fournir des informations vraiment utiles pour le processus de synchronisation. Par exemple, la préparation d'une recette de cuisine est fréquemment accompagnée d'un commentaire audio reprenant les instructions de préparation ou pour le moins reprenant des mots clés issus de la recette en fonction de l'étape atteinte. Nous imaginons qu'il pourrait être possible, dans certains cas, de procéder à alignement du texte prononcé sur la bande son la procédure écrite. Après la synchronisation vidéo/texte, l'alignement textuel pourrait dans certains cas affiner les limites des segments vidéo.

Pour aller plus loin, certains sons peuvent aussi renforcer les segmentations en action ou en répétitions.

6.4 Les conditions

Problématique :

Le contenu textuel peut contenir des indicateurs de contrôle comme les conditions. Plusieurs enregistrements vidéo peuvent avoir été produits sur la base d'un même contenu textuel. Chacun d'eux peut correspondre à la réalisation d'un chemin différent dans la procédure. Dans notre travail, nous avons considéré uniquement le cas d'un chemin textuel et d'une vidéo qui contient sa réalisation.

Propositions :

Dans une solution complète de synchronisation de contenu, nous imaginons un système capable de segmenter une vidéo donnée et d'identifier automatiquement le chemin réalisé dans cette vidéo, avant de synchroniser les segments détectés avec les instructions du chemin correspondant. Dans un système indépendant du type des contenus, il est impossible d'identifier ce chemin sans éléments pour reconnaitre ou comparer les actions réalisées.

Scénario 1 : Nous imaginons un ensemble d'enregistrements vidéo qui contient les réalisations de tous les chemins possible dans un contenu textuel. Cet ensemble est divisé en plusieurs sous-ensembles, dont chacun continent les réalisations du même chemin. Les vidéos de chaque sous-ensemble sont bien synchronisées avec les instructions du chemin correspondant. À l'arrivée d'un nouvel enregistrement vidéo à traiter, nous lançons la synchronisation de cette vidéo avec chaque chemin textuel possible. À partir de la matrice de confiance de chaque chemin, nous obtenons une séquence des segments vidéo. D'autre part, les vidéos dans le corpus ont également des séquences de segments synchronisées avec les chemins correspondants. La comparaison entre les segments de la vidéo et les segments de la séquence de chaque vidéo dans le corpus peut venir alimenter la décision. Dans le cas où les segments de la vidéo sont plus similaires (une mesure de similarité doit être identifiée) avec les segments des séquences des vidéos d'un sous-ensemble donné qu'avec les autres, cela peut être un indice pour identifier le chemin correspondant.

Scénario 2 : Dans certain cas, les chemins textuels peuvent être distingués par un critère donné comme l'existence d'une boucle additionnelle, la ré-réalisation d'une instruction, l'existence d'une instruction spéciale (ex. demander de l'opérateur de ne plus bouger pendant une minute), etc…. La détection de ce critère peut être utile pour identifier le chemin dans la vidéo analysée.

6.5 Distribution bidimensionnelle

Problématique :

Dans le système de synchronisation, nous avons utilisé une distribution verticale des coefficients de confiance. Cela indique qu'une limite textuelle (associée à une colonne) peut correspondre à un point temporel situé dans le voisinage du point automatiquement associé. Le cas inverse peut aussi être considéré.

Par exemple, un point temporel sélectionné par l'outil de segmentation peut correspondre à une limite textuelle dans le voisinage de la limite textuelle initialement associée par l'outil. Pour cela, nous pensons qu'il est utile de distribuer les coefficients sur les colonnes situées dans le voisinage de la limite textuelle associée par l'outil.

Proposition :

Au lieu d'une distribution verticale et donc monodimensionnelle de coefficients, nous supposons qu'il pourrait être judicieux d'opter pour une distribution bidimensionnelle qui répartirait le coefficient "1" de chaque association (ligne/colonne : L /C) d'une manière horizontale et verticale.

6.6 Extraction des associations

Problématique :

Dans le système de synchronisation, les associations sont extraites à partir de la matrice finale de confiance en effectuant une recherche récursive. Durant cette recherche, nous recalculons des sous matrices (haut-gauche et bas-droite) après chaque extraction d'une association. Mais des contradictions peuvent survenir, comme par exemple, dans le cas où la troisième limite textuelle est associée à la trame numéro 1. Dans un tel cas, nous essayons alors de trouver des correspondances aux deux premières limites textuelles parmi des trames situées avant la trame numéro 1 !

Proposition :

Une solution possible à ce problème, peut-être de détecter une contradiction et de modifier une décision prise à une étape antérieure, puis de vérifier si cette modification résout le problème. Or, la question est de savoir quelle est la décision qui doit être remise en question et quelle alternative doit être choisie. Dans notre travail, nous n'avons pas proposé de solution pour traiter les contradictions. Voilà un scenario possible :

Une décision prise à une étape donnée correspond à la sélection des points temporels parmi tous les points proposés par un outil donné, et à leur association à un

ensemble des limites textuelles. En pratique, les points sélectionnés sont les points les plus fiables parmi ceux fournis par l'outil (les plus haut votes). Une contradiction atteinte à une étape donnée indique qu'une décision fausse a été prise à une étape précédente.

À ce moment, nous proposons de retourner une étape en arrière et de changer la décision prise à cette étape en sélectionnant un autre ensemble de points, puis de reprendre la procédure d'extraction à partir de cette étape. Si la contradiction persiste après ce changement de décision, nous retournons cette fois à l'antépénultième étape, et ainsi de suite.

La mise en œuvre de cette proposition doit conduire à déterminer une stratégie permettant de sélectionner judicieusement un ensemble alternatif de points lorsque les premiers, considérés comme étant les plus fiables, conduisent à une solution.

6.7 Affiner les limites des segments synchronisés

Problématique :

Pour extraire les associations dans la matrice finale de confiance, nous avons proposé une méthode récursive qui extrait les coefficients maximaux dans la matrice l'un après l'autre. Les limites des segments sont ensuite associées aux limites des instructions d'une manière indépendante. Les associations ne considèrent pas la nature des limites qui peuvent être des débuts ou des fins. La seule contrainte considérée par notre méthode est que le début d'un segment se trouve bien évidemment avant sa fin. Cela n'empêche pas la détection potentielle d'un segment de 1 seule trame !

Proposition :

Dans notre méthode, après l'identification de la colonne et la ligne qui contient le coefficient maximal dans la matrice finale de confiance, nous relançons la recherche dans deux zones : en haut à droite et en bas à gauche. Dans une version modifiée de notre méthode nous pourrions chercher à distinguer le type de limite et à considérer l'association extraite : début ou fin d'un segment. Nous pourrions alors exploiter une contrainte qui contrôlerait la taille minimale d'un segment associé. La définition de cette valeur, dépend fortement du type des actions réalisées dans la vidéo.

6.8 Reconnaissance d'actions

Problématique :

Comme on l'a déjà mentionné, le système de synchronisation proposé est un système général et indépendant des contenus tant qu'ils respectent les hypothèses

formulées. Ce système pourra être personnalisé et adapté pour synchroniser un type spécialisé de contenus l'exécution d'une recette de cuisine. Dans ce cas, d'autres travaux de reconnaissance d'actions prédéfinies et propres à ce domaine (hacher, mélanger, étaler, etc… pour la cuisine), permettrait d'affiner les associations entre les segments vidéo et les instructions correspondantes.

Propositions :

Scenario 1 : Un outil capable de reconnaitre une action prédéfinie peut ajouter de nouvelles informations liées à la segmentation des actions qui n'agissent pas sur la scène (*ex*. salutation par la main, étreinte, *etc.)* qui sont indétectables par notre outil de segmentation d'actions. Après la détection de l'action, cet outil pourra alors construire sa propre matrice de confiance en associant les segments détectés aux instructions qui correspondent à cette action.

Scenario 2 : Si après la première application de notre système, un outil d'identification détecte une action prédéfinie dans un segment mais qui est associé par notre système à une autre instruction donnée, nous imaginons que la réexécution du système de synchronisation en prenant cette information en considération pourra affiner les résultats.

6.9 Vidéo et texte structurés

Problématique :

Le texte est souvent organisé, structuré en paragraphes, avec des numéros de paragraphes, des titres, etc. Dans certains cas, la vidéo peut elle-même être structurée (les vidéos de la recette des crêpes sont montées ou éditées pour en limiter la durée). On peut alors sans doute utiliser ces éléments d'édition dans la stratégie de mise en correspondance.

Proposition :

On pourrait étudier dans quelle mesure les limites des plans, ou l'information amenée par des incrustations de texte (pour lesquels il existe de nombreux outils d'analyse automatique), pourraient permettre de préciser la localisation temporelle de début ou de fin d'action.

Une fois ce système enrichi par une certaine diversité d'informations, son champ d'application pourra être élargi : maintenance technique, domaine médical, domaine artistique, *etc.* Par exemple, il pourrait être utilisé comme un système de vérification qui examinerait l'exécution du texte prescriptif pour mesurer le niveau de respect de la procédure et savoir si l'opérateur a bien complété sa tâche, si l'infirmière a bien effectué les soins, si le danseur a bien respecté la chorégraphie, *etc.*

BIBLIOGRAPHIE

(1) Cheveigné, A., Kawahara, H.
YIN, a fundamental frequency estimator for speech and music
The Journal of the Acoustical Society of America, volume 111, issue 4, pp. 1917-1930, 2002
DOI: 10.1121/1.1458024

(2) Haidar, S., Joly, P., Chebaro, B.
Mining for video production invariants to measure style similarity
International Journal of Intelligent Systems, Volume 21, Issue 7, pp. 747–763, 2006
DOI: 10.1002/int.20158

(3) Takahashi, M., Irie, K.. Terabayashi, K.. Umeda, K.
Gesture Recognition Based on the Detection of Periodic Motion
International Symposium on Optomechatronic Technologies (ISOT), 2010, pp. 1 – 6, 2010
DOI: 10.1109/ISOT.2010.5687352

(4) CHAMAND, B.
Analyse de gestes répétés à l'aide des matrices de similarité
Rapport de stage dans le cadre du 1ère cycle ingénieur à l'école d'ingénieur interne (UPSSITECH – IRIT), 2015

(5) Barrios, J. M., Bustos, B.,
Competitive content-based video copy detection using global descriptors.
Multimedia Tools and Applications, volume 62, issue 1, pp. 75 – 110, 2013
DOI: 10.1007/s11042-011-0915-x

(6) Covell, M., Baluja, S., Fink, M.
Advertisement Detection and Replacement using Acoustic and Visual Repetition
IEEE 8th Workshop on Multimedia Signal Processing, pp. 461 – 466, 2006
DOI: 10.1109/MMSP.2006.285351

(7) Cutler, R., Davis, L.
Robust real-time periodic motion detection, analysis, and applications
IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 22, issue 8, pp. 781 – 796, 2000.
DOI: 10.1109/34.868681

(8) Döhring, I., Rainer, L.
Mining TV broadcasts for recurring video sequences.
Proceedings of the ACM International Conference on Image and Video Retrieval, pp. 1 – 8, 2009.
DOI: 10.1145/1646396.1646432

(9) Duygulu, P., Chen, M.-y., Hauptmann, A.
Comparison and combination of two novel commercial detection methods
IEEE International Conference on Multimedia and Expo, volume 2, pp. 1267 – 1270, 2004.
DOI: 10.1109/ICME.2004.1394454

(10) Esmaeili, M., Fatourechi, M., Ward, R.
A Robust and Fast Video Copy Detection System Using Content-Based Fingerprinting
IEEE Transactions on Information Forensics and Security, volume 6, issue 1, pp. 213 – 226, 2010.
DOI: 10.1109/TIFS.2010.2097593

(11) Hampapur, A., Hyun, K., Bolle, R. M.
Comparison of sequence matching techniques for video copy detection
Proceedings SPIE 4676, Storage and Retrieval for Media Databases, 4676, pp. 194-201, 2002.
DOI: 10.1117/12.451091

(12) Hauptmann, A., Witbrock, M.
Story segmentation and detection of commercials in broadcast news video
IEEE International Forum on Research and Technology Advances in Digital Libraries, pp. 168 – 179, 1998.
DOI: 10.1109/ADL.1998.670392

(13) Indyk, P., Iyengar, G., Shivakumar, N.
Finding pirated video sequences on the Internet. Stanford University
Technical report, Stanford University, pp. 117 – 128, 1999.

(14) Jiang, Z., Lin, Z., Davis, L. S.
Recognizing Human Actions by Learning and Matching Shape-Motion Prototype Trees
IEEE 12th International Conference on Computer Vision, pp. 444 – 451, 2009.
DOI: 10.1109/ICCV.2009.5459184

(15) Joly, A., Buisson, O., Frelicot, C.
Content-Based Copy Retrieval Using Distortion-Based Probabilistic Similarity Search
IEEE Transactions on Multimedia, , volume 9, issue 2, pp. 293 – 306, 2007
DOI: 10.1109/TMM.2006.886278

(16) Laptev, I.
On Space-Time Interest Points
International Journal of Computer Vision, volume 64, issue 2-3, pp. 107 – 123, 2005
DOI: 10.1007/s11263-005-1838-7

(17) Laptev, I., Lindeberg, T.
Velocity adaptation of space-time interest points
Proceedings of the 17th International Conference on Pattern Recognition, volume 1, pp. 52 – 56, 2004
DOI: 10.1109/ICPR.2004.1334003

(18) Laptev, I., Belongie, S., Perez, P., Wills, J.
Periodic motion detection and segmentation via approximate sequence alignment
International Conference on Computer Vision. 1, pp. 816-823. 2005.
DOI: 10.1109/ICCV.2005.188

(19) Law-To, J., Chen, L., Joly, A., Laptev, I., Buisson, O., Gouet-Brunet, V., Stentiford, F.
Video copy detection: a comparative study
ACM international conference on Image and video retrieval, pp. 371 – 378, 2007.
DOI: 10.1145/1282280.1282336

(20) Li, Y., Zhang, D., Zhou, X., Jin, J. S.
A confidence based recognition system for TV commercial extraction
The nineteenth conference on Australasian database, volume 75, pp. 57 – 64, 2008.

(21) Lienhart, R., Kuhmunch, C., Effelsberg, W.
On the detection and recognition of television commercials
Proceedings of IEEE International Conference on Multimedia Computing and Systems, pp. 509 – 516, 1997.
DOI: 10.1109/MMCS.1997.609763

(22) Oostveen, J., Kalker, T., & Haitsma, J.
Feature extraction and a database strategy for video fingerprinting
Recent Advances in Visual Information Systems of the series Lecture Notes in Computer Science, volume 2314, pp. 117 – 128, 2002.
DOI: 10.1007/3-540-45925-1_11

(23) Saracoğlu, A., Esen, E., Ates, T.K., Acar, B.O., Zubari, U., Ozan, E.C., Ozlap, E., Alatan, A.A. Ciloglu, T.,
Content Based Copy Detection with Coarse Audio-Visual Fingerprints
Seventh International Workshop on Content-Based Multimedia Indexing, pp. 213 – 218, 2009.
DOI: 10.1109/CBMI.2009.12

(24) Shivadas, A., & Gauch, J.
Real-Time Commercial Recognition Using Color Moments and Hashing
Fourth Canadian Conference on Computer and Robot Vision, pp. 465 – 472, 2007.
DOI: 10.1109/CRV.2007.53

(25) Tong, X., Duan, L., Xu, C., Tian, Q., Lu, H., Wang, J., & Jin, J.
Periodicity Detection of Local Motion
IEEE International Conference on Multimedia and Expo, pp. 650 – 653, 2005.
DOI: 10.1109/ICME.2005.1521507

(26) Vlachos, M., Philip, S. Y., Castelli, V.
On Periodicity Detection and Structural Periodic Similarity
Proceedings of the Fifth SIAM International Conference on Data Mining, pp. 449-460, 2005.

(27) Cour, T., Jordan, C., Miltsakaki, E., Taskar B.
Movie/Script: Alignment And Parsing Of Video And Text Transcription
Proceedings of the 10th European Conference on Computer Vision – part IV, pp. 158 – 171, 2008.
DOI: 10.1007/978-3-540-88693-8_12

(28) Wang, F., Ngo, C. W., Pong, T. C
Synchronization Of Lecture Videos And Electronic Slides By Video Text Analysis
Proceedings of the eleventh ACM international conference on Multimedia, pp. 315 – 318, 2003.
DOI: 10.1145/957013.957080

(29) Chen, Y., Heng, W. J.
Automatic Synchronization Of Speech Transcript And Slides In Presentation
IEEE Proceedings of the 2003 International Symposium on Circuits and Systems, volume 2, pp. 568 – 571, 2003.
DOI: 10.1109/ISCAS.2003.1206037

(30) Amir, A., Niblack, C.W., Pass, N.J., Petkovic, D., Ponceleon, D.B., Srinivasan, S., Syeda-Mahood, T.F.
System and method for linking an audio stream with accompanying text material
U.S. Patent, US6636238, 2003.

(31) Ellozy, H. A., Kanevsky, D., Kim, M. Y., Nahamoo, D., Picheny, M. A., & Zadrozny, W. W.
Automatic indexing and aligning of audio and text using speech recognition
U.S. Patent, US5649060, 1997.

(32) Takebayashi, Y., Tsuboi, H., Kanazawa, H.
A Robust Speech Recognition System Using Word-spotting with Noise Immunity Learning
IEEE International Conference on Acoustics, Speech, and Signal Processing, volume 2, pp. 905 – 908, 1991.
DOI: 10.1109/ICASSP.1991.150486

(33) Gaidon, A., Marszalek, M., Schmid, C.
Mining visual actions from movies
Proceedings of the British Machine Vision Conference, pp. 125.1 – 125.11, 2009.
DOI: 10.5244/c.23.125

(34) Hauptmann, A. G., Witbrock, M. J.
Story segmentation and detection of commercials in broadcast news video
Proceedings. IEEE International Forum on Research and Technology Advances in Digital Libraries, pp. 168 – 179, 1998.
DOI: 10.1109/ADL.1998.670392

(35) Everingham, M., Sivic, J., Zisserman, A.
"Hello! My name is... Buffy" – Automatic Naming of Characters in TV Video
Proceedings of the British Machine Vision Conference, pp. 92.1-92.10, 2006.
DOI: 10.5244/c.20.92

(36) Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.
Learning realistic human actions from movies
IEEE Conference on Computer Vision and Pattern Recognition, pp. 1 – 8, 2008.
DOI: 10.1109/cvpr.2008.4587756

(37) Naturel, X., Gravier, G., Gros, P
Fast Structuring of Large Television Streams Using Program Guides
Adaptive Multimedia Retrieval: User, Context, and Feedback of the series Lecture Notes in Computer Science, volume 4398, pp. 222-231, 2007.
DOI: 10.1007/978-3-540-71545-0_17

(38) Guyon, I., Athitsos, V., Jangyodsuk, P., Escalante, H.J.
The ChaLearn gesture dataset (CGD 2011)
Machine Vision and Applications, volume 25, issue 8 , pp. 1929 – 1951, 2014.
DOI: 10.1007/s00138-014-0596-3

(39) Molina, J., Pajuelo, J. A., Escudero-Viñolo, M.., Bescós, J., Martínez, J. M.
A natural and synthetic corpus for benchmarking of hand gesture recognition systems
Machine Vision and Applications, volume 25, issue 4, pp. 943-954, 2013.
DOI: 10.1007/s00138-013-0576-z

(40) Over, P., et *al.*.
TRECVID 2015 - An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics
Proceedings of TRECVID 2015, 2015.
http://www-nlpir.nist.gov/projects/tvpubs/tv15.papers/tv15overview.pdf

(41) Vacavant, A., Chateau, T., Wilhelm, A., Lequièvre, L.
A Benchmark Dataset for Outdoor Foreground/Background Extraction
Computer Vision - ACCV 2012 Workshops of the series Lecture Notes in Computer Science, volume 7728, pp. 291 – 300, 2013.
DOI: 10.1007/978-3-642-37410-4_25

(42) Foote, J., Cooper, M.,
Visualizing musical structure and rhythm via self-similarity
Proceedings of the 2001 International Computer Music Conference, pp. 419 – 422, 2001

(43) Roger B., Dannenberg, Ning, Hu,
Pattern Discovery Techniques for Music Audio
Journal of New Music Research, volume 32, issue 3, pp. 153 – 163, 2003
DOI: 10.1076/jnmr.32.2.153.16738

(44) Lie, Lu, Wang, M., Hong-Jiang, Z.
Repeating pattern discovery and structure analysis from acoustic music data
Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval,

pp. 275 – 282, 2004
DOI: 10.1145/1026711.1026756

(45) Kim, K., Chalidabhongse, T. H., Harwood, D., Davis, L.
Real-time foreground–background segmentation using codebook model
Real-Time Imaging, volume 11, issue 3, pp. 172 – 185, 2005
DOI: 10.1016/j.rti.2004.12.004

(46) Barnich, O., Van Droogenbroeck, M.
ViBe: A Universal Background Subtraction Algorithm for Video Sequences
IEEE Transactions on Image Processing, volume 6, pp. 1709 – 1724, 2010
DOI: 10.1109/TIP.2010.2101613

(47) Maddalena, L., Petrosino, A.,
A Self-Organizing Approach to Background Subtraction for Visual Surveillance Applications
IEEE Transactions on Image Processing, volume 17, issue 7, pp. 1168 – 1177, 2008
DOI: 10.1109/TIP.2008.924285

(48) Elhabian, S.Y., El-Sayed, K.M., Ahmed, S.H.
Moving Object Detection in Spatial Domain using Background Removal Techniques - State-of-Art
Recent Patents on Computer Science, volume 1, issue 1, pp. 32 – 54, 2008

(49) Ambata L.U., Caluyo F.S.
Background change detection using wavelet transform.
TENCON 2012 - 2012 IEEE Region 10 Conference, pp. 1 – 6, 2012
DOI: 10.1109/tencon.2012.6412298

(50) Cucchiara, R., Grana, C., Piccardi, M., Prati, A.
Detecting moving objects, ghosts and shadows in video streams
Pattern Analysis and Machine Intelligence, volume 25, issue 10, pp. 1337 – 1342, 2003
DOI: 10.1109/tpami.2003.1233909

(51) Elgammal, A., Duraiswami, R., Harwood, D., Davis, L.S.
Background and foreground modeling using nonparametric kernel density estimation for visual surveillance
Proceedings of the IEEE, volume 90, issue 7, pp. 1151 – 1163, 2002
DOI: 10.1109/JPROC.2002.801448

(52) Fihl, P., Corlin, R., Park, S., Moeslund, T.B., Trivedi, M.M.
Tracking of individuals in very long video sequences
Advances in Visual Computing Lecture Notes in Computer Science, volume 4291, pp. 60 – 69, 2006
DOI: 10.1007/11919476_7

(53) Geng L., Xiao, Z.T.
Real Time Foreground-Background Segmentation Using Two-Layer Codebook Model
2011 International Conference on Control, Automation and Systems Engineering (CASE), volume 1,

pp. 1 – 5, 2011
DOI: 10.1109/ICCASE.2011.5997546

(54) Gibbins, D., Newsam, G.N., Brooks, M.J.
Detecting suspicious background changes in video surveillance of busy scenes
Proceedings 3rd IEEE Workshop on Applications of Computer Vision, 1996. WACV '96., pp. 22 – 26, 1996
DOI: 10.1109/acv.1996.571990

(55) Gong, Y., Sin, L.T., Chuan, C.H., Zhang, H., Sakauchi, M.
Automatic Parsing of TV Soccer Programs
International Conference on Multimedia Computing and Systems, volume 1, pp. 0167, 1995
DOI: 10.1109/MMCS.1995.484921

(56) Ilyas, A., Scuturici, M., Miguet, S.
Real Time Foreground-Background Segmentation Using a Modified Codebook Model
Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance, pp. 454 – 459, 2009
DOI: 10.1109/AVSS.2009.85

(57) Leykin, A., Ran, R., Hammoud, R.
Thermal-visible video fusion for moving target tracking and pedestrian classification
IEEE Conference on Computer Vision and Pattern Recognition, volume 1, pp. 1 – 8, 2007
DOI: 10.1109/CVPR.2007.383444

(58) Radke, R.J., Andra, S., Al-Kofahi, O., Roysam, B.
Image Change Detection Algorithms - A Systematic Survey
IEEE Transactions on Image Processing, volume 14, issue 3, pp. 294 – 307, 2005
DOI: 10.1109/tip.2004.838698

(59) Rui, Y., Gupta, A., Acero, A.
Automatically Extracting Highlights for TV Baseball Programs
Proceedings of the eighth ACM international conference on Multimedia, volume 8, pp. 105 – 115, 2000
DOI: 10.1145/354384.354443

(60) Sigari, M.H., Fathy, M.
Real-time Background Modeling/Subtraction using Two-Layer Codebook Model
Proceedings of the International MultiConference of Engineers and Computer Scientists, volume 1, pp. 717 – 720, 2008

(61) Stauffer, C., Grimson, E.
Adaptive background mixture models for real-time tracking
IEEE Computer Society Conference on Computer Vision and Pattern Recognition, volume 2, pp.

252, 1999
DOI: 10.1109/CVPR.1999.784637

(62) Sudhir, G., Lee, J.C.M., Jain, A.K.
Automatic classification of tennis video for high-level content-based retrieval
IEEE International Workshop on Content-Based Access of Image and Video Database, pp. 81 – 90, 1998
DOI: 10.1109/caivd.1998.646036

(63) Thome, N., Miguet, S.,
A robust appearance model for tracking human motion
IEEE Conference on Advanced Video and Signal Based Surveillance, pp.528 – 533, 2005
DOI: 10.1109/AVSS.2005.1577324

(64) Zhang, D., Chang, S.
Event detection in baseball video using superimposed caption recognition
Proceedings of the tenth ACM international conference on Multimedia, pp. 315 – 318, 2002
DOI: 10.1145/641007.641073

(65) MathWorks laboratories
Scene Change Detection System (vipscenechange)
Computer Vision System Toolbox in MATLAB (R2013a), 2013
http://fr.mathworks.com/help/vision/examples/scene-change-detection.html Accessed 1 December 2014

(66) Xuan, Mo., Monga, V., Bala, R., Zhigang, Fan.
Adaptive Sparse Representations for Video Anomaly Detection
IEEE Transactions on in Circuits and Systems for Video Technology, volume 24, issue 4, pp. 631 – 645, 2014
DOI: 10.1109/TCSVT.2013.2280061

(67) Quintas, J., Khoshhal, K., Aliakbarpour, H., Dias, J.
Human, Behaviour Analysis: Fall Detection, 2012
https://www.xsens.com/customer-cases/human-behaviour-analysis-fall-detection/.

(68) YingLi, Tian, Feris, R.S., Haowei, Liu, Hampapur, A., Ming-Ting Sun
Robust Detection of Abandoned and Removed Objects in Complex Surveillance Videos
IEEE Transactions on in Systems, Man, and Cybernetics, Part C: Applications and Reviews, volume 41, issue 5, pp. 565 – 576, 2011
DOI: 10.1109/TSMCC.2010.2065803

(69) Chun-Ku, Lee, Meng-Fen, Ho, Wu-Sheng, Wen, Chung-Lin, Huang,
Abnormal Event Detection in Video Using N-cut Clustering
International Conference on in Intelligent Information Hiding and Multimedia Signal Processing,

pp. 407 – 410, 2006
DOI: 10.1109/IIH-MSP.2006.265028

(70) Uğur Töreyin, B., Dedeoğlu, Y., Enis Çetin, A.
HMM Based Falling Person Detection Using Both Audio and Video
Computer Vision in Human-Computer Interaction, Volume 3766 of the series Lecture Notes in Computer Science, pp. 211 – 220, 2005
DOI: 10.1007/11573425_21

(71) Rougier, C., Meunier, J., St-Arnaud, A., Rousseau, J.
Fall Detection from Human Shape and Motion History Using Video Surveillance
21st International Conference on in Advanced Information Networking and Applications Workshops, volume 2, pp. 875 – 880, 2007
DOI: 10.1109/AINAW.2007.181

(72) Text mining. Encyclopédie Wikipedia. https://en.wikipedia.org/wiki/Text_mining

(73) Nadeau, D., Sekine, S.
A survey of named entity recognition and classification
Lingvisticae Investigationes, Volume 30, issue 1, pp. 3 – 26, 2007
DOI: 10.1075/li.30.1.03nad

(74) Zhou, G., Su, J.
Named Entity Recognition Using an HMM-based Chunk Tagger
Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 473–480, 2002
DOI: 10.3115/1073083.1073163

(75) Florian, R., Ittycheriah, A., Jing, H., Zhang, T.
Named Entity Recognition Through Classifier Combination
Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL, volume 4, pp. 168 – 171, 2003
DOI: 10.3115/1119176.1119201

BIOGRAPHIE

Publications

Journaux scientifiques:

Wehbe, H., Haidar, B., Joly, P., “Action boundaries detection in a video”, Multimedia Tools and Applications, Volume 75, Issue 14, pp. 8239-8266, 2016, DOI: 10.1007/s11042-015-2748-5m

Haidar, B., Chebaro, B., Wehbe, H., “An Evaluation Model for the Chains of Distributed Multimedia Indexing Tools Respecting User Preferences”. Journal of Theoretical and Applied Information Technology, Vol. 13, Issue 2, pp. 139-147, 2010.

Conférences:

Wehbe, H., Haidar, B., Joly, P., “Automatic Detection of Repetitive Actions in a Video”, 13th International Workshop on Content-Based Multimedia Indexing, pp. 1-6, 2015, DOI: 10.1109/CBMI.2015.7153605.

Enseignement supérieurs

Depuis 2007 : Enseignant contractuel dans les laboratoires du département des mathématiques appliquées. Les 500 heures du contrat sont réalisées comme des sessions de TP et de TD des cours suivants :

Matières	Niveau
1. TP réseaux et systèmes	M2
2. Multimédia	M1
3. Programmation réseau	M1
4. Interconnexion des réseaux	L3 et M1
5. Réseaux informatiques (1 et 2)	L2 et L3
6. Système d’exploitation (1 et 2)	L2 et L3
7. Structures de données	L2
8. Base de données	L3
9. Programmation logique	L2
10. Programmation impérative	L2
11. Algorithmique et programmation	L1

Encadrement

Depuis 2008 : Encadrant de plusieurs étudiants – à chaque année – durant leurs Projets de « Base de données » et « Structure de données ».

Stage de Master de recherche

Titre de Master : Coopération dans les Sciences du Traitement de l'Information.

Université : Université Libanaise en partenariat avec l'Université Paul Sabatier à Toulouse.

Année universitaire : 2005/2006.

Encadrant : Bassem Haidar.

Titre de travail : Évaluation d'une chaîne d'outils d'indexation multimédia.

Résumé de travail : Avec l'augmentation de nombre des documents multimédia utilisés, le besoin d'identifier et de trouver facilement ces documents à partir des index a devenu très grand, d'où le domaine d'indexation multimédia. Les index sont générés en partant d'un document multimédia primitif, par un programme appelé outil d'indexation. Mais certains index ne peuvent pas être générés directement en utilisant un seul outil, d'où le besoin de chaîner un ensemble d'outils pour atteindre notre but. Généralement, ce chaînage s'effectue manuellement mais récemment un algorithme permettant d'automatiser ce processus a été proposé, mais sans classifier les chaînes résultants. Notre objectif dans ce sujet est de définir et mettre en place des critères visant à évaluer et classifier les chaînes d'outils d'indexation trouvées par cet algorithme.

Synchronisation Automatique d'un Contenu Audiovisuel avec un Texte qui le Décrit

Abstract

We address the problem of automatic synchronization of an audiovisual content with a procedural text that describes it. The strategy consists in extracting pieces of information about the structure from both contents, and in matching them depending on their types.

We propose two video analysis tools that respectively extract:

→ Limits of events of interest using an approach inspired by dictionary quantization.

→ Segments that enclose a repeated action based on the YIN frequency analysis method.

We then propose a synchronization system that merges results coming from these tools in order to establish links between textual instructions and the corresponding video segments. To do so, a "Confidence Matrix" is built and recursively processed in order to identify these links in respect with their reliability.